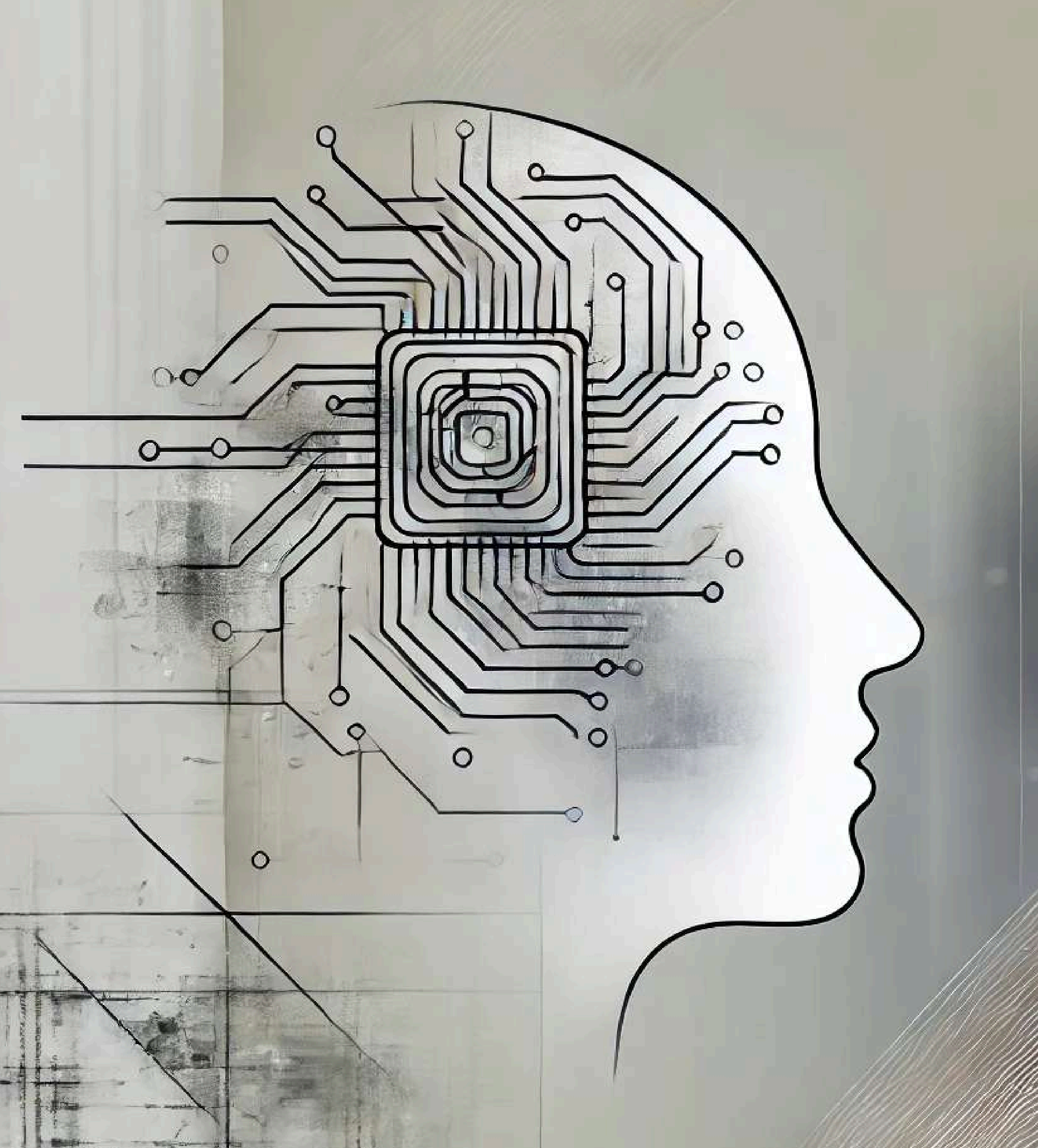


Inteligencia Artificial

Los 3 pasos para formarte desde cero y transformar tu futuro

2025 v1.0

Gabriel Pierobon

El contenido de este libro, y fundamentalmente lo que se encuentra dentro de recuadros fue desarrollado con asistencia de modelos de inteligencia artificial, específicamente Claude de Anthropic, y editado por el autor entre los meses de abril de 2024 y febrero de 2025. Las imágenes, salvo que se indique lo contrario, fueron creadas utilizando DALL-E de OpenAI. Todo el contenido fue supervisado, revisado y aprobado por el autor. Este libro ha sido creado con fines educativos y de divulgación, sin ningún tipo de ánimo de lucro. A través de estas páginas, el autor ha querido ofrecer su propia visión del presente y futuro de la Inteligencia Artificial, entretejiendo conceptos técnicos con las experiencias y aprendizajes de su trayectoria profesional y sus investigaciones en el campo. El objetivo del autor no es posicionarse como un experto técnico, sino servir de puente para que conceptos complejos sean más accesibles y comprensibles para todos aquellos interesados en adentrarse en el fascinante mundo de la IA. Habiendo identificado una escasez de material comprehensivo sobre este tema en español, el autor busca hacer este conocimiento accesible a aquellas personas que, no dominando el inglés, se han visto limitadas en su acceso a recursos de calidad sobre Inteligencia Artificial.

Título del Libro: "Inteligencia Artificial: Los 3 pasos para formarte desde cero y transformar tu futuro"

Autor: Gabriel Pierobon

Este libro no pretende ser un compendio exhaustivo de todos los conceptos y técnicas de la Inteligencia Artificial, sino una guía práctica y accesible que cubre los fundamentos esenciales. Está especialmente diseñado tanto para profesionales que buscan actualizarse en el campo de la IA como para personas que están dando sus primeros pasos en esta disciplina. El enfoque elegido prioriza la claridad y la aplicabilidad práctica por sobre la exhaustividad técnica, proporcionando una base sólida para que el lector pueda seguir profundizando en las áreas que más le interesen.

Primera edición: v1.0.0 02/2025

Publicado por Gabriel Pierobon

Madrid, España

Índice de contenidos

Notas del autor

Sentado en la mesa de mi habitación de hotel en Múnich una mañana muy soleada de un Mayo que no te puedo explicar lo intenso que viene siendo. En un par de horas me toca ir a participar de un *workshop*, uno de esos donde se ponen a prueba las habilidades de comunicación (entre vos y yo, quisiera mejorar mucho en ese aspecto). Me gusta aprender cosas nuevas, hay que aprender cosas nuevas; el mundo cambia, evoluciona y nosotros tenemos que hacerlo con él. Por eso acepto el desafío con ilusión y un poco de ansiedad.

Saber comunicar es una habilidad extremadamente importante. Sobre todo hacerlo en una época tan marcada por contrastes, donde por un lado cada día cuesta más explicar el avance de los desarrollos tecnológicos, pero por el otro tenemos nuevos productos como ChatGPT que nos asisten y nos ayudan a hacer simple lo complejo. La semana anterior me tocó poner a prueba estas habilidades de comunicación en un viaje de trabajo por Estados Unidos, más precisamente en la ciudad de Boston visitando a dos de nuestros clientes más importantes. ¿De qué hablamos con ellos?

De cambiarles la forma de pensar sus negocios gracias a la **inteligencia artificial**.

En mi empresa estamos convencidos de que les vamos a cambiar la vida (laboral) a todos los que confíen en nuestro producto y nuestra misión es ayudarles a adoptar esta tecnología que tiene el potencial para revolucionar al mundo. Dejar de hablar de posibilidades y planes futuros y poner en sus manos (literalmente, por que es una aplicación principalmente móvil) un producto listo para usar.

Meterme en el campo de la inteligencia artificial me abrió puertas al mundo y yo ahora lo tomo como algo normal; creo que a estas alturas ya me acostumbré. Es mi trabajo. Me muevo de Madrid (donde vivo actualmente) a Barcelona, a Nueva York, a Boston, a Chicago, a Múnich… Parece no detenerse ahí. A veces hago el ejercicio de frenar un poco, encontrar una pausa para pensar cómo llegué a esto. Hago la conexión de los eventos más importantes que terminaron conmigo en este preciso lugar. Es importante hacer ese ejercicio, no vivir siempre con la inercia.

¿Los primeros 30 años de mi vida? En casa, Buenos Aires. Ahí me formé y senté mis bases.

¿Como programador? No…

¿Como ingeniero en sistemas? No…

¿Como matemático? Tampoco…

No no… Licenciado en Economía.

Economista de la Universidad de Buenos Aires (UBA) y posteriormente MBA[1] porque confiaba que era la herramienta que iba a propulsar mi carrera empresarial (me sirvió, no diré que no).

De todas maneras, el tipo de carrera que estaba haciendo llegó a un punto en el que empezó a aburrirme, pero por sobre todas las cosas comenzó también a generarme un cierto nivel de preocupación. Poco a poco empecé a sentir que lo que hacía cada día, mis habilidades, mi *skill-set* (me gusta el término en inglés[2]) poco iban de la mano con lo que se demandaría en el mercado laboral en un futuro *no tan lejano*.

¿Tengo un skill-set del futuro?

¿Vendrán buenas empresas a buscarme?

Peor aún, *¿seguirá existiendo el trabajo que hago ahora mismo en tres, cinco, siete años?*

Ese tipo de preguntas me daban vueltas por la cabeza constantemente. Mi puesto de *Director de Control de Gestión* era bastante bueno y bien compensado, pero yo dependía de él y de ninguna manera quería sentirme atado a ningún puesto de trabajo, ni mucho menos vivir con miedo a perderlo. Definitivamente, y por sobre todas las cosas, no quería imaginarme una situación donde ver el mundo avanzar como un espectador y no *ser parte* de eso.

Entonces me pregunté un día como hoy hace ocho años. *¿Hacia dónde se mueve el mundo? ¿Qué habilidades se necesitan no hoy, no mañana, pero dentro de diez años, de quince, de veinte?*

Mientras me hacía estas preguntas, algún día del año 2017, uno de los conceptos más mencionados en el *mainstream* empresarial era *Big Data* (¡por dios cómo cambia el mundo que esa expresión ya casi ni se usa!).

Pero en aquel entonces, era **BIG DATA**.

[1] UCEMA, Buenos Aires, Argentina
[2] Traduzcamos a *conjunto de habilidades*

Decidí apuntarme en un programa en la Universidad CEU San Pablo en Madrid. Sesenta horas de clases solamente, para ***enterarme un poco*** al menos. Un programa corto llamado *"Programa Avanzado en Big Data y Visual Analytics"*. Para ganar ***contexto***.

😱 **Me cambió la vida.**

Te preguntarás, *¿el programa te enseñó todo lo que necesitabas?*

Mmmm… No.

¿Hacer el programa te permitió conseguir un puesto para empezar a trabajar de eso aunque sea como un junior?

Tampoco.

¿Aprendiste a liderar proyectos de Big Data?
¿Te enseñaron a programar en Python o R?
¿Entendiste todo lo que hace falta sobre arquitecturas cloud en AWS, Microsoft Azure o Google Cloud?
¿Te explicaron en detalle los algoritmos más importantes de machine learning?

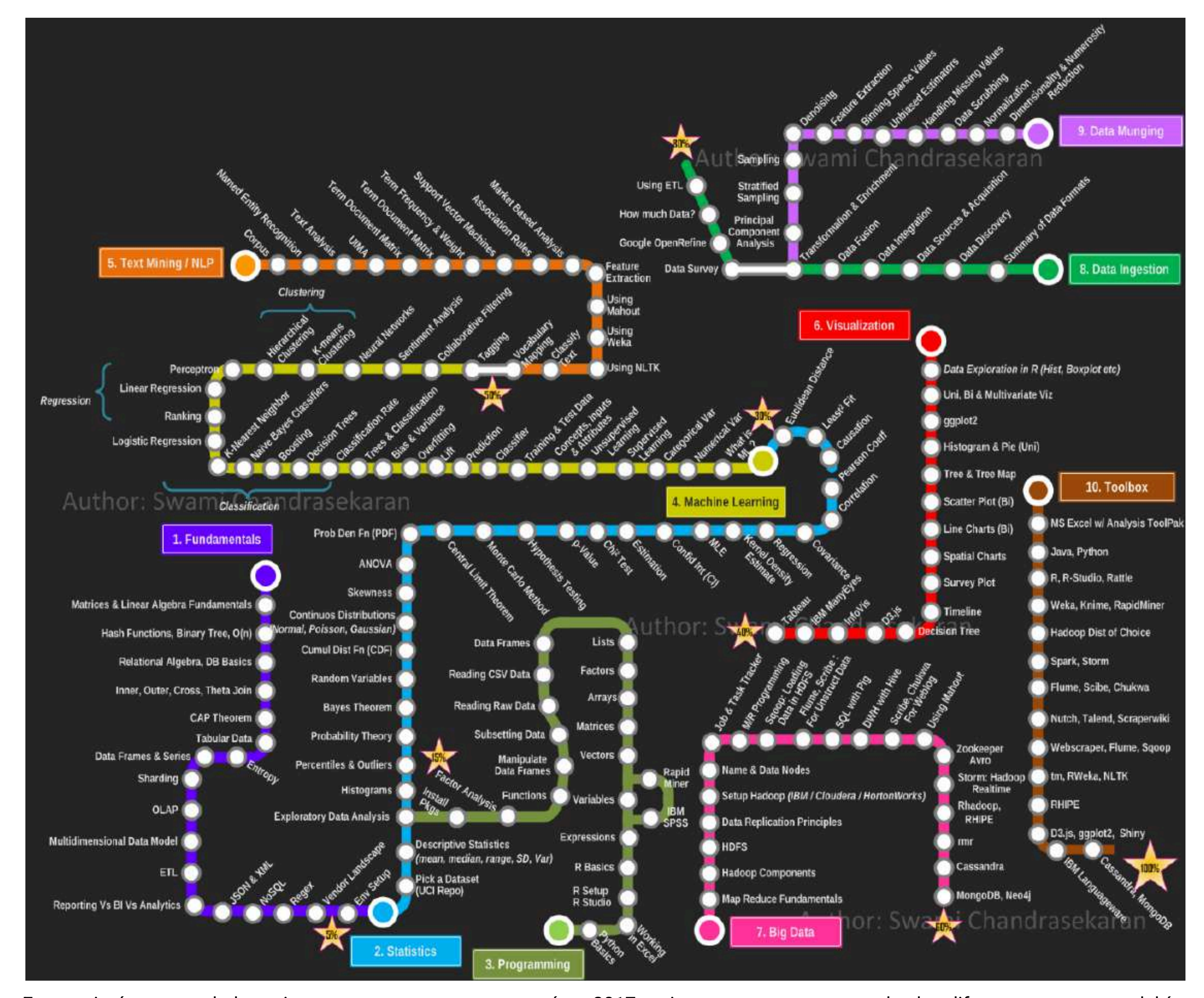
Apenas, algo, no y para nada.

Lo que **SÍ** hizo ese programa por mí fue **darme contexto**. El famoso ***me abrió la cabeza***.

Contexto de cómo el mundo estaba afrontando los desafíos ligados con el procesamiento de datos (todavía se hablaba más de *data analytics / data science / data mining* que de *inteligencia artificial*). **Contexto** de cómo los datos comenzaban cada vez más a propulsar soluciones y productos basados en algoritmos que *aprenden* de ellos y pueden asistir en la toma de decisiones. **Contexto** de cómo el mundo se inclinaba cada vez más hacia la *personalización* (recomendación de contenido, para básicamente generar más *engagement* con los clientes), y, posiblemente sin mencionarlo explícitamente ninguna vez, **contexto** acerca de la revolución que vendría con la ***inteligencia artificial*** tarde o temprano.

No necesité más que eso.

Ese ***contexto*** me permitió trazar un *mapa* entre las habilidades que yo tenía hasta ese entonces, y las que necesitaría para dominar ese futuro dentro de tres, cinco, siete años.



En esta imágen, una de los primeros esquemas que encontré en 2017 en internet para conocer todos los diferentes temas que debía aprender si quería convertirme en un Data Scientist. Este esquema lo utilicé para *medir* la distancia entre lo que sabía y lo que no. Está desactualizado en 2024 así que no te sugiero que le prestes demasiada atención. Son 10 temáticas: 1- Fundamentos, 2- Estadística, 3- Programación, 4- Machine Learning, 5- NLP, 6- Visualización de datos, 7- Big Data, 8- Ingestión de Datos, 9- Transformación de Datos, 10- Herramientas. (Autor: Swarni Chandrasekaran)

Como **economista** me sentía bastante lejos de ese objetivo, pero sabía que tenía la base perfecta para perseguirlo. Es muy importante dar valor a la carrera realizada, al *background* de cada uno, sin importar cual. Tu *formación* se trata de aprender a resolver problemas, utilizar herramientas y aplicar pensamiento crítico. Provengas de la rama que provengas, te aseguro que tenés una base sólida y lista para abordar cualquier desafío. Yo, si bien hasta ese momento siempre había trabajado en finanzas, lo veía como un trabajo que me permitía hacer lo que más me gustaba: analizar datos. De analizar datos a dominar la inteligencia artificial había lógicamente un camino largo a recorrer.

En mi caso personal, recorrer el camino trazado y llegar a uno de los *buenos* destinos dependió del empeño, las posibilidades que se me fueron planteando, las prioridades[3] y el siempre presente componente aleatorio. Después, como todo en la vida, hay que ponerle sacrificio y esperar que la suerte acompañe un poco también. Con esa suerte y con ese empeño, ahora estoy feliz y orgulloso de estar trabajando todos los días *tocando* la **inteligencia artificial,** descubriendo cómo aplicarla para resolver problemas y hacer eso que siempre disfruté: analizar datos.

En el día a día, casi todo el tiempo ignoro este recorrido, un poco me olvido, y es que vos y yo somos iguales, vivimos en el presente, en lo que estamos haciendo ahora mismo y siempre se mira hacia adelante. Si me detengo a pensarlo, me enorgullece haber llegado a uno de esos posibles *buenos* destinos y estar ahora acá escribiendo este libro para contarte cómo es posible hacerlo realidad.

Quiero que este libro sea para vos algo como el *programa* que te conté antes, ese de sesenta horas ***que me abrió la cabeza***.

Por eso, no te voy a enseñar a programar, no te voy a enseñar a entrenar una red neuronal, no te voy a enseñar a usar los LLM ni a levantar un servidor en AWS para ejecutar un *contenedor* de Docker. **Lo que pretendo hacer es ponerte en contexto**. Te voy a hablar de mi interpretación del contexto actual de la inteligencia artificial y trataré de convencerte de que **ahora** es el momento de aprender estas habilidades, que aún estás a tiempo, pero que si te interesa tenés que aprovechar este preciso momento. Te voy a explicar todos los temas básicos de la inteligencia artificial y también te voy a brindar recursos para que comiences tu camino y adoptes ciertas rutinas para que el aprendizaje sea diario, sencillo y continuo.

El libro se organiza en **tres grandes pasos** que tenés que dar[4]. No te los voy a adelantar, pero creeme que cuando hayas dado estos tres pasos, vas a haber entrado de lleno en el maravilloso mundo de la inteligencia artificial y comenzado a transformar tu futuro.

[3] Nada es gratis. El costo de oportunidad que nos enseñan en economía está siempre presente y mis decisiones también me hicieron perder cosas que me importaban.

[4] Como habrás notado en el índice de contenidos, en términos de longitud, el segundo paso es "infinitamente" más extenso que los otros dos, pero el diseño me pareció el más apropiado para el objetivo de este libro ;)

Introducción

Para empezar, dejame que capture tu imaginación por un momento.

Imaginate por un segundo, tan solo un segundo, que retrocedés en el tiempo…

… ¿Cuánto?

Viajemos hacia aquellos primeros días del **Bitcoin**, cuando todavía era una novedad que pocos conocían y muchos menos entendían. ¿Te imaginás haber invertido en ese entonces, comprendiendo el espectacular impacto y valoración que terminaría teniendo en el futuro?

¿No? Hagamos algunos números.

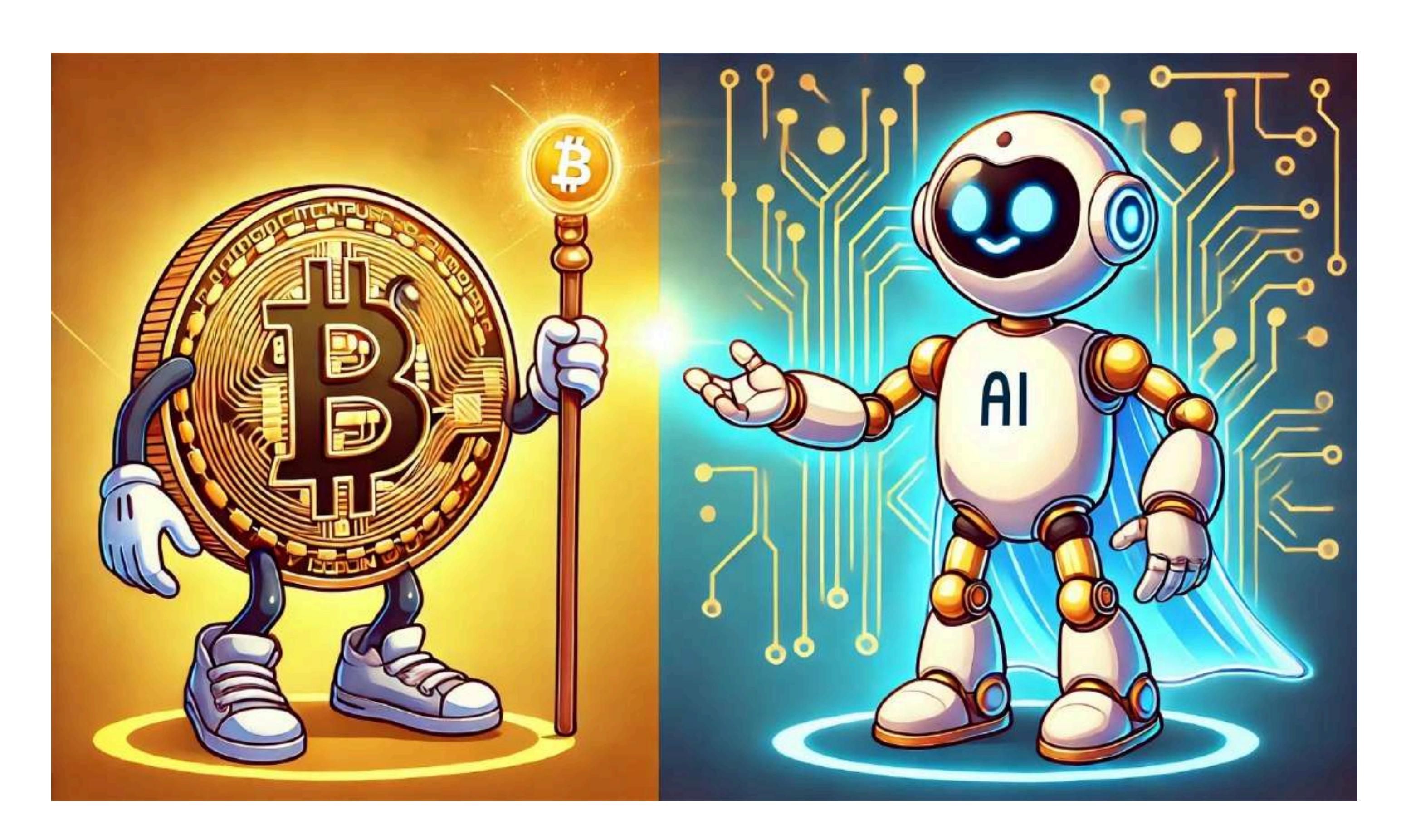


Si en algún momento del año 2015 hubieras comprado 10 Bitcoin cuando su precio era aproximadamente U$S 320 cada uno, totalizaría una inversión de U$S 3,200 y hoy (escribo esto en abril de 2024) tendrías una ganancia neta de U$S 663,574 (seiscientos sesenta y tres mil quinientos setenta y cuatro dólares!!!). En enero de 2019, si hubieras invertido U$S 38,000 comprando 10 Bitcoin a U$S 3,800 cada uno, tu ganancia ahora sería de $628,774. Llegados a este punto consideremos una inversión más accesible, digamos la compra de 1 solo Bitcoin

en septiembre de 2020 por U$S 10,750. Tu inversión ahora valdría U$S 66,677 lo que representa una ganancia significativa de U$S 55,927 en menos de cuatro años. Y finalmente, si hubieras decidido entrar en el juego un poco más tarde, comprando 1 Bitcoin en octubre de 2023 a U$S 26,000 hoy verías una ganancia de U$S 40,677.[5]

Este es solo un ejemplo, no estoy asumiendo que ninguno de nosotros tuvo ni tiene necesariamente los recursos para hacer esa inversión, ni mucho menos que tiene un nivel de aversión al riesgo tal que la fundamente. En mi caso particular, por ejemplo, he de admitir que tengo un nivel de aversión al riesgo muy alto, no suelo invertir en activos de alta volatilidad. De todas formas el ejemplo no es tanto para hablar de criptomonedas sino que me es útil como un testimonio del poder de la innovación tecnológica, de estar en el lugar correcto en el momento adecuado y de reconocer una gran oportunidad capaz de transformar tu futuro.

Pero en este libro vine a hablarte de inteligencia artificial, así que rápidamente movámonos hacia ese espacio.



[5] Los valores presentados son aproximados y tienen fines ilustrativos. Los precios históricos pueden variar según la fuente y el mercado de criptomonedas es altamente volátil.

Dejame decirte que la ***inteligencia artificial***[6] representa ahora mismo **esa oportunidad comparable a las primeras etapas del Bitcoin** para aquellos dispuestos a sumergirse y participar activamente. No como un *activo financiero*, pero sí como una inversión en *capital humano*, una inversión en tus habilidades que puede darte un retorno muy alto en tu futura carrera laboral.

Lógicamente, los que tomaron el riesgo quince años atrás de invertir en estas habilidades[7], hoy seguramente gozan del *florecimiento* del campo, pero también los que entramos ocho años atrás (solo ocho!!!) vemos un nivel de oportunidades laborales increíbles y sentimos una cierta libertad de saber que podemos elegir donde queremos trabajar[8]. Esto es recién el comienzo del auge de la IA y mi opinión es que estas oportunidades crecerán cada vez más, pero tengo el deber de comunicarte que el momento de formarte en IA es **ahora**!

Así como con el Bitcoin, donde una inversión temprana podía traducirse en ganancias significativas a futuro, meterse y formarse en el mundo de la IA **ahora**, cuando está recién comenzando a moldear activamente industrias y crear nuevas oportunidades te posicionará para capitalizar en el futuro. Esto no necesariamente significa invertir dinero, sino invertir tiempo, energía, esfuerzo y atención para entender la IA, aprender a trabajar con ella y aplicarla en tu campo de interés, profesión o emprendimiento.

No hay almuerzo gratis nos decía un profesor de Finanzas del MBA, y como economista que soy, siempre voy a intentar ponerlo en términos de *costo de oportunidad*. Al fin y al cabo, si decidís invertir tus recursos en aprender IA, estarás dejando de lado otras alternativas de inversión igualmente viables. Uno de mis objetivos con este libro será darte una cierta seguridad de que si decidís seguir este camino, será altamente rentable para tu carrera laboral.

Pero bien, ¿por qué digo que **ahora** es el momento ideal? Porque la IA, lejos de ser una promesa futurista, ya es una realidad palpable que está modelando nuestro presente. Si alguna vez pensaste que el futuro estaría dominado por la IA, bueno, ese futuro puede estar comenzando ahora mismo. Las herramientas, el conocimiento y las oportunidades están al alcance de la mano, esperando que alguien con suficiente visión las aproveche. Y mientras algunos siguen debatiendo si este es el camino a seguir, otros ya están trazando su ruta hacia el éxito.

---

[6] A lo largo de este libro me voy a referir a la inteligencia artificial por sus siglas IA o AI de *artificial intelligence*, indistintamente.
[7] Riesgo en el sentido de que no necesariamente era una industria en auge
[8] No literalmente, pero me refiero a esa noción de *si no me gusta, sé que puedo cambiar de trabajo*

Curiosamente, incluso este libro que estás leyendo es un testimonio del poder de la IA. Parte de este contenido ha sido generado con la asistencia de inteligencia artificial[9], un claro ejemplo de cómo esta tecnología puede ser una herramienta invaluable para amplificar nuestras capacidades creativas y productivas. Si la idea de que una máquina pueda ayudar a escribir un libro te resulta extraña o incluso preocupante, te invito a reconsiderarlo. La IA no viene a reemplazarnos, sino a complementarnos, a permitirnos alcanzar nuevas alturas de innovación y creatividad. La posibilidad de que alguien como yo, que no es un maestro de la escritura y la comunicación pueda proponer un producto así me parece un ejemplo maravilloso de cómo esta tecnología amplifica capacidades. Imaginate tener que ser un experto en Física Cuántica y además un genio de la comunicación y el *copywriting* para poder transmitir tus mensajes. Eso sería digno de admiración.

Mi propio viaje hacia el dominio de la IA no fue un camino de rosas. En el momento que decidí aprenderlo, aposté todo[10]. Dejé un cómodo puesto de Dirección y estuve siete meses sin trabajo dedicado cien por ciento a aprender todo lo que pudiera sobre este mundo tan fascinante. Durante esos siete meses me despertaba a las 6am y terminaba mi día a las 9pm y entre medio consumía infinidad de contenido sobre IA y Data Science. Quería entenderlo todo, quería dominarlo todo y lo quería hacer rápidamente. Ésta no es mi recomendación para todo el mundo, yo tenía la posibilidad de hacerlo[11] y me salió bien. De hecho, en este libro te invito a tomar un camino diferente. En lugar de hacer un *all-in* como hice yo y dedicarle a la IA el 100%, hacerlo de una forma más pausada, pero constante y en base a rutinas. No va a hacer falta que dejes tu trabajo, pero a partir de ahora deberías asignarle parte de tu más preciado recurso… tu tiempo.

A lo largo de este libro además te voy a mostrar que, independientemente de tu formación o experiencia previa, hay un lugar para vos en el mundo de la IA. Juntos vamos a desmitificar esta tecnología, explorar sus aplicaciones prácticas y, lo más importante, aprender cómo podés comenzar a ser parte de esta revolución.

Sin más preámbulos, te invito a que demos el primer paso juntos…

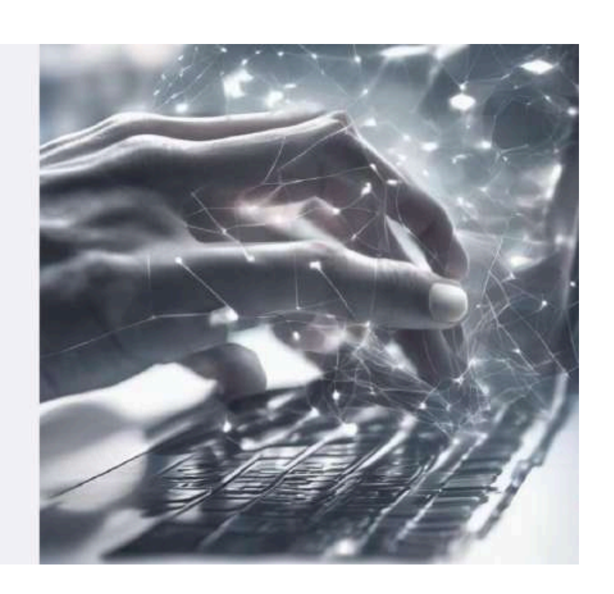
---

[9] Contenido que aparezca dentro de recuadros en el libro, el cual ha sido generado con asistencia parcial de IA.
[10] Incluso a pesar de mi ya comentada alta aversión al riesgo
[11] Había ahorrado suficiente para darme el lujo.

Primer Paso: entrar AHORA

La Oportunidad de la IA

La Inteligencia Artificial (IA) ofrece una oportunidad sin precedentes para empresas de todos los tamaños. Si bien la adopción de la IA aún está en una fase temprana, este es el momento ideal para comenzar a incorporarla y aprovechar su potencial transformador.



¡Felicitaciones! El primer paso, el más importante de todos, ¡lo acabás de dar!

Claro, al interesarte por este libro y haber llegado a esta precisa página. ¿Terminamos el capítulo acá entonces? Eso no sería una buena idea realmente. Mejor dejame que te cuente por qué el momento de meterse en el mundo de la inteligencia artificial es ahora[12].

¿Por qué el momento de meterse en la IA es ahora?

Considerá esto: el valor que se le atribuye al mercado global de la inteligencia artificial, que ya era significativo en 2024, está proyectado para más que triplicarse en solo cinco años, pasando de de USD $136 billones en 2022 a USD $407 billones en 2027[13]. Este crecimiento esperado no solo resalta la importancia económica de la IA sino también su impacto transformador en múltiples industrias.

Números así podrían hacerte pensar que *ya es tarde*, que en un mercado así las empresas ya están totalmente sumergidas en el viaje de la IA, que ya existen sobrados profesionales capacitados en esta tecnología mucho tiempo atrás y que posiblemente ya dejaste pasar el momento. Entonces surgen las dudas y la sensación de que *esto no es para mi*.

Dejame decirte que eso ¡no es para nada así! Por que el momento ideal para meterse es:



Prestá atención a este *tweet* y vas a entender por qué te digo que el momento es ahora.

[12] Vamos a considerar que por *ahora* nos referimos a un período comprendido entre 2025 y 2026. Si estás leyendo este libro a partir de 2027 en adelante, el libro es igualmente útil, pero el capítulo debería llamarse Primer Paso: Viaja al pasado y entrá AHORA

[13] https://connect.comptia.org/blog/artificial-intelligence-statistics-facts

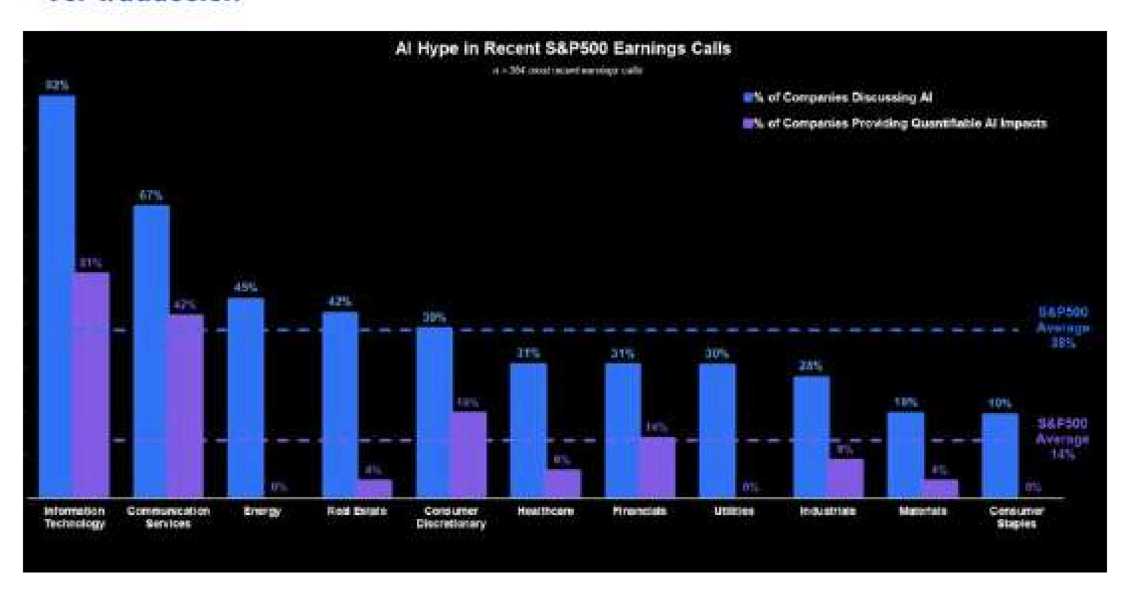




~40% of S&P 500 companies talked about AI.
~13% actually were able to report value.

We used AI Agents to read every earnings call this season. 

Ver traducción



Tweet de Hebbia AI el 9 de Mayo de 2024.
40% de las compañías del S&P[14] 500 hablan sobre IA.
13% realmente pudo reportar valor.

Esto es increíblemente impactante. Solo una de cada diez de las empresas más valiosas del mundo reportaba en 2024 valor a través de la aplicación de inteligencia artificial.

Esto quiere decir que las empresas a nivel global aún se encuentran en una fase de adopción donde por un lado reconocen que integrar la IA en sus procesos es tan inevitable como necesario, pero por el otro, aún así no logran un impacto concreto.

¿Qué implicaría conseguir un impacto concreto con la integración de la inteligencia artificial en procesos empresariales? Algunas posibilidades:

- 💡 **Aumento en la eficiencia operativa:** implementación de soluciones de IA que automaticen tareas rutinarias y reduzcan los tiempos de procesamiento, mejorando así la productividad general de la empresa.

[14] El S&P 500, o Standard & Poor's 500, es un índice bursátil que mide el rendimiento de 500 de las empresas más grandes y cotizadas en las bolsas de valores de Estados Unidos. Considerado uno de los indicadores más representativos de la salud del mercado de acciones estadounidense, el S&P 500 abarca una amplia gama de industrias y sectores. Su selección se basa en criterios como la capitalización de mercado, liquidez y representación sectorial, y es utilizado por inversores para evaluar el rendimiento del mercado de valores y tomar decisiones de inversión.

- 🧠 **Mejora en la toma de decisiones:** uso de análisis predictivo y modelos de *machine learning* para proporcionar *insights* más precisos, permitiendo decisiones estratégicas basadas en datos y minimizando el riesgo.
- 💰 **Optimización de costos:** reducción significativa de costos operativos a través de la automatización de procesos y la optimización de recursos, generando ahorros financieros tangibles.
- 🚀 **Innovación en productos y servicios:** desarrollo de nuevos productos y servicios basados en tecnologías de IA, que puedan abrir nuevas oportunidades de mercado y mejorar la competitividad.
- 😊 **Mejora en la experiencia del cliente:** implementación de sistemas de IA, como chatbots o asistentes virtuales, que mejoren la atención al cliente, ofrezcan respuestas rápidas y personalizadas, que resulte en una mayor satisfacción y lealtad del cliente.

La mayoría de estas empresas, en realidad, se quedan en *pruebas de concepto (POCs)* y poco o nada es puesto en producción, pero no por que no sepan cómo ni por que se estén equivocando, sino por que **es parte del camino de adopción** por el que una tecnología tan disruptiva y con tanto potencial debe transitar.



Un ejemplo histórico similar se puede observar con la adopción de la electricidad a finales del siglo XIX y principios del XX. En sus primeras etapas, muchas empresas experimentaron con la electricidad a través de proyectos piloto y pruebas en instalaciones limitadas. Inicialmente, la electricidad se usaba principalmente para iluminación, pero la transición completa de los sistemas de energía basados en vapor y otros métodos tradicionales tomó tiempo. Al igual que con la inteligencia artificial hoy en día, no se trataba de una falta de comprensión o de errores en la implementación, sino de un proceso necesario de adaptación y aprendizaje. Las empresas tuvieron que descubrir cómo integrar la electricidad de manera efectiva en sus operaciones diarias, identificar las oportunidades más prometedoras y superar desafíos técnicos y económicos. Con el tiempo, la electricidad se convirtió en una fuerza transformadora que revolucionó la industria y la vida cotidiana, del mismo modo que se espera que la IA lo haga en la actualidad.

Y para vos eso es una ventaja, una ventana de oportunidad única en el tiempo que te indica que **todavía es temprano para aprender sobre IA**, incorporar estas herramientas a tus conocimientos y aprovechar los beneficios económicos que supone.

Repasemos:

Aún Estamos en la Superficie

Pocas Empresas Implementan IA

Sólo el 13% de las compañías del S&P 500 han logrado reportar un valor concreto a través de la IA. Esto muestra que la mayoría aún está en una fase de adopción temprana.

Enorme Potencial de Crecimiento

Se espera que el mercado global de IA se más que triplique en los próximos cinco años, pasando de $136 billones en 2022 a $407 billones en 2027.

Transformación de Industrias

La IA tiene el potencial de contribuir con $15.7 trillones a la economía global para 2030, redefiniendo industrias enteras y creando nuevas oportunidades de trabajo.



¿Qué se espera en unos pocos años? La IA tiene el potencial de contribuir con USD 15.7 trillones a la economía global para 2030, redefiniendo industrias enteras y creando nuevas oportunidades de trabajo. A pesar de la preocupación sobre la eliminación de puestos de trabajo debido a la automatización, la realidad es que la IA creará 97 millones de nuevos empleos para 2025[15], superando la cifra de empleos que se prevé que elimine.

Imaginate el impacto: sectores como el minorista, la salud, la manufactura y la banca están sólo empezando a rascar la superficie de lo que la IA puede ofrecer, desde mejorar la productividad del empleado en un 40% hasta generar billones de dólares en valor agregado para la economía global.

[15] https://explodingtopics.com/blog/ai-statistics

Aprovechá la Ventana de Oportunidad



Adopción Temprana

La mayoría de las empresas aún se encuentran en las primeras etapas de integración de la IA, lo que significa que hay una ventana de oportunidad para aquellos que tomen la delantera.



Creación de Empleo

Contrario a los temores, la IA creará 97 millones de nuevos empleos para 2025, superando la cantidad de puestos que se prevé que elimine.



Redefinición de Industrias

Sectores como el minorista, la salud y la banca están empezando a ver el enorme potencial de la IA para mejorar la productividad y generar valor.



Ventana de Oportunidad

Este es el momento ideal para aprender sobre IA e incorporar estas herramientas a tu negocio o carrera, aprovechando los beneficios económicos que supone.

La revolución de la IA recién comienza

Lo que es cierto es que la Inteligencia Artificial viene recorriendo un camino que va desde los márgenes de la ciencia ficción hasta el centro del escenario tecnológico mundial en la actualidad. La irrupción de OpenAI con ChatGPT a fines de 2022 ha sido un catalizador reciente en este proceso, redefiniendo nuestras expectativas sobre lo que la IA es capaz de hacer. Al momento de escribir estas líneas, las empresas que están desarrollando los modelos de lenguaje más avanzados son OpenAI con GPT, Anthropic con Claude, Google con Gemini, Mistral AI con Mixtral y Meta con Llama. Luego tenemos a Google, Microsoft, AWS y Apple introduciendo IA en toda su gama de productos tanto de hardware como software.

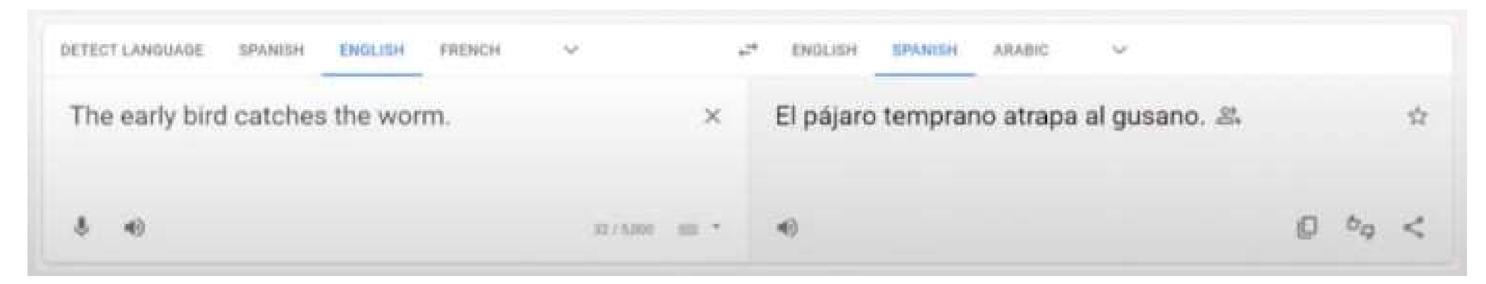
Tengo dos recuerdos grabados que me hacen poner en perspectiva el progreso exponencial que vivimos actualmente con la IA y darme cuenta que hasta hace muy poco, todo esto parecía imposible. Vas a tener que perdonarme por unos minutos, por que voy a saltarme muchos temas básicos para hablar de desarrollos más recientes en temas de IA, pero es para ilustrar un punto muy importante de este *Primer Paso*. Te prometo que luego vamos a volver a ver estos temas explicados de forma clara y sencilla.

El primero de estos dos recuerdos tiene que ver con el día en el que finalmente me di cuenta lo espectacular que se había vuelto **el traductor de Google**, que dejó de ser una herramienta

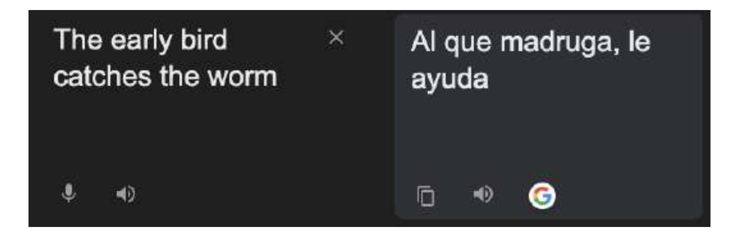
poco confiable y casi inútil para traducir textos largos del español al inglés y viceversa, y pasó a ser prácticamente perfecto. Casi por arte de magia.

Acá van un ejemplo tonto pero que ilustra el cambio:

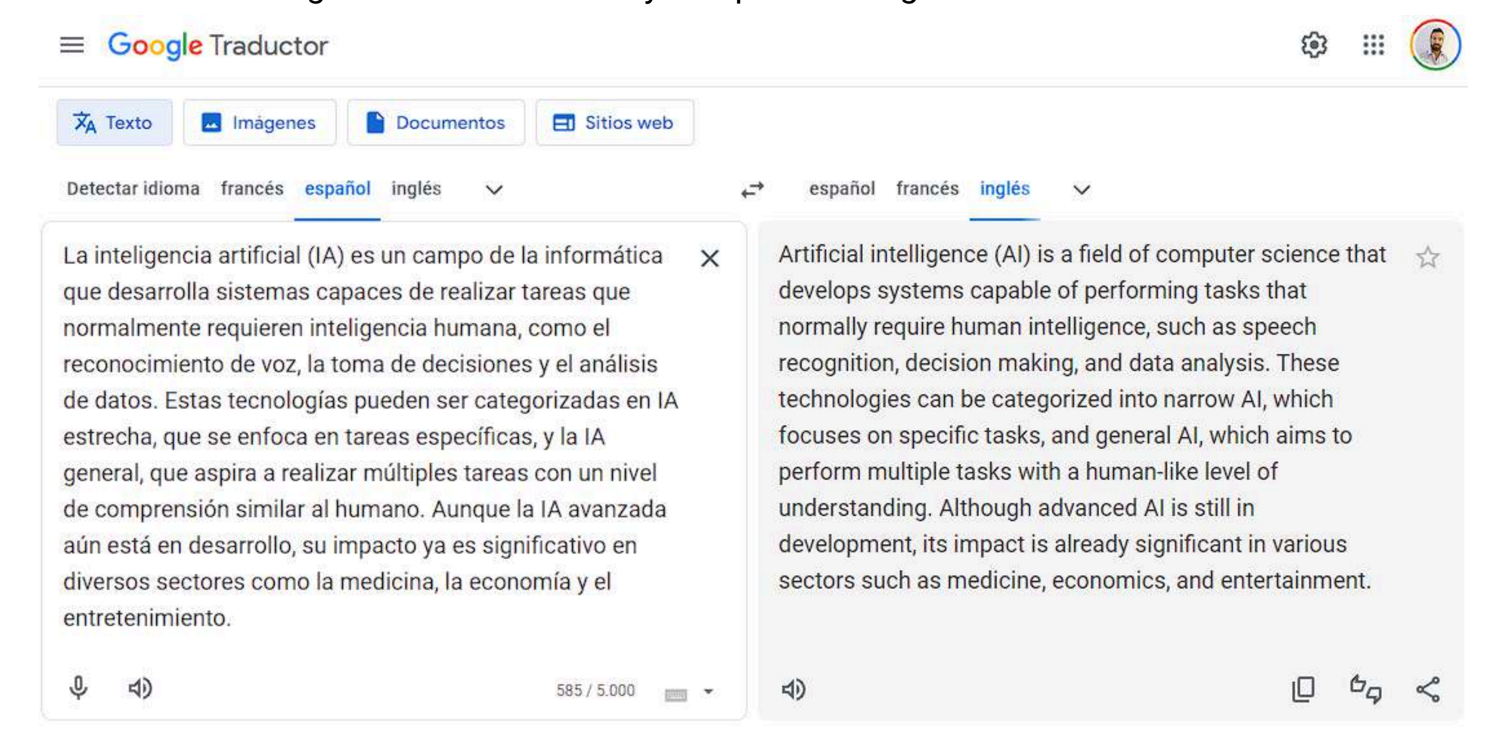
Antes:



Ahora:



Aunque pueda parecer una tontería, en ese momento me sorprendió muchísimo el avance tecnológico que permitió semejante desarrollo. De repente, largos textos eran perfectamente traducidos sin ningún error de contexto y con perfecta lógica.



Por debajo de semejante cambio, estaba el detonante del increíble crecimiento de la inteligencia artificial que vemos hoy en día: el ***Transformer***. En el caso concreto del traductor de Google, permitió la integración del modelo BERT a finales de 2019, lo que mejoró notablemente la precisión de las traducciones. Además, BERT también empezó a potenciar el **buscador** de Google, revolucionando la forma en que obtenemos información de internet. Este cambio fue tan rotundo que transformó por completo la experiencia de los usuarios al realizar búsquedas, haciendo el proceso más intuitivo y eficiente.

Es tan importante la aparición de BERT que está considerado uno de los desarrollos más importantes de la historia de la Inteligencia Artificial. Acá te dejo un artículo muy interesante sobre los 10 hitos de la IA de Google, uno de los cuales es el propio BERT.

https://tekiosmag.com/2023/09/28/google-celebra-sus-25-anos-aqui-los-10-hitos-del-desarrollo-de-su-inteligencia-artificial/

Pero, ¿qué es BERT exactamente y por qué fue tan importante?



BERT (**B**idirectional **E**ncoder **R**epresentations from **T**ransformers) se basa en la arquitectura **Transformer**, introducida por Vaswani et al. en 2017[16]. BERT es un modelo generativo de lenguaje natural, es decir, es capaz de escribir texto de forma secuencial (palabra por palabra) de la misma

[16] https://arxiv.org/pdf/1706.03762 El mítico paper “Attention is all you need”

forma que vemos hacerlo hoy en día a ChatGPT.

A diferencia de los modelos tradicionales de procesamiento del lenguaje natural (NLP) que procesaban el texto de manera secuencial hasta ese momento, BERT utiliza un enfoque *bidireccional*, lo que significa que considera tanto el contexto anterior como el posterior de una palabra en una oración. Esta arquitectura permite a BERT capturar relaciones más complejas entre las palabras, mejorando significativamente el rendimiento en diversas tareas de NLP como la clasificación de texto, el reconocimiento de entidades y la respuesta a preguntas.

Antes del *Transformer*, las máquinas no eran muy buenas para entender el significado de oraciones largas. No podían *ver* las relaciones entre palabras que estaban muy separadas. El *Transformer* mejoró enormemente esto y se convirtió en la base de los sistemas de inteligencia artificial generativa y comprensión del lenguaje más impresionantes de la actualidad.

«El Transformer ha revolucionado lo que significa para las máquinas traducir, resumir textos, responder preguntas e incluso generar imágenes, así como también la robótica»[17]

Este modelo se entrenó inicialmente con grandes cantidades de texto (como todo el contenido de Wikipedia) en dos tareas: predecir palabras ocultas dentro de una oración y determinar si una oración sigue a otra en un texto. Esta fase de *pre-entrenamiento* permite que BERT adquiera un conocimiento *profundo* del lenguaje, que luego puede aplicarse a tareas específicas como clasificar textos, responder preguntas o analizar sentimientos con solo un poco de ajuste adicional.

Entrenamiento de BERT

1. Predicción de palabras enmascaradas (Masked Language Modeling)

BERT aprende a predecir palabras que han sido enmascaradas aleatoriamente en una oración.

> Entrada: "El [MASK] es azul y tiene nubes blancas."
>
> BERT predice: "El **cielo** es azul y tiene nubes blancas."

En la práctica, BERT mejoró significativamente la precisión y la capacidad de las aplicaciones de procesamiento de lenguaje natural. Como ya vimos, esto se traduce en motores de búsqueda más precisos, asistentes virtuales más inteligentes y herramientas de traducción más efectivas, haciendo que las interacciones entre humanos y máquinas sean más naturales y eficientes.

17 https://tekiosmag.com/2023/09/28/google-celebra-sus-25-anos-aqui-los-10-hitos-del-desarrollo-de-su-inteligencia-artificial/



2. Predicción de la siguiente frase (Next Sentence Prediction)

BERT aprende a predecir si una frase B es la continuación lógica de una frase A.

> Frase A: "El gato está sentado en el sofá."
>
> Frase B: "Está durmiendo plácidamente."
>
> BERT predice: **Es continuación**

> Frase A: "El gato está sentado en el sofá."
>
> Frase B: "Los pingüinos viven en la Antártida."
>
> BERT predice: **No es continuación**

Es así como entonces el traductor de Google se convirtió en una herramienta poderosísima, al punto en que cada vez que me tocaba traducir algún documento, podía confiar en hacerlo semi-automáticamente con él, cosa que antes me resultaba impracticable.

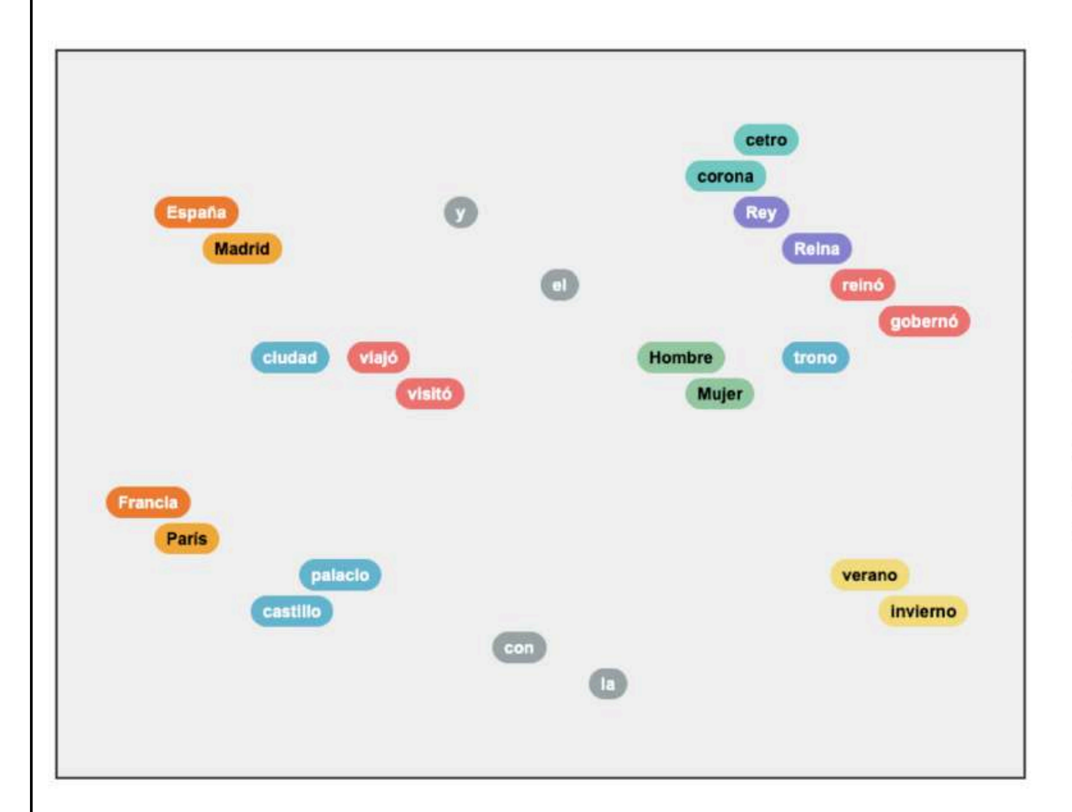


Y esto concluye el primero de mis dos recuerdos de *“Wow! La inteligencia artificial es cosa seria!*

El segundo recuerdo, y tal vez mi tópico favorito en temas de *procesamiento de lenguaje natural* fue cuando conocí la *magia* de los **Word Embeddings**.

Los word embeddings son una técnica que convierte palabras en una especie de "coordenadas" dentro de un *espacio vectorial*, similar a un mapa. Esto ayuda a las computadoras a entender mejor el significado de las palabras y cómo se relacionan entre sí. Antes de que existieran los word embeddings, las palabras se representaban de manera muy simple, sin captar las conexiones entre ellas. Con esta técnica, palabras que tienen significados parecidos se colocan cerca unas de otras en este mapa imaginario.

Vamos a ver esto en detalle más adelante en el libro pero por ahora en el siguiente ejemplo podés ver una representación de cómo las palabras de un texto se asocian en ese *espacio vectorial*. Notá como las palabras que se relacionan aparecen más cerca agrupadas.

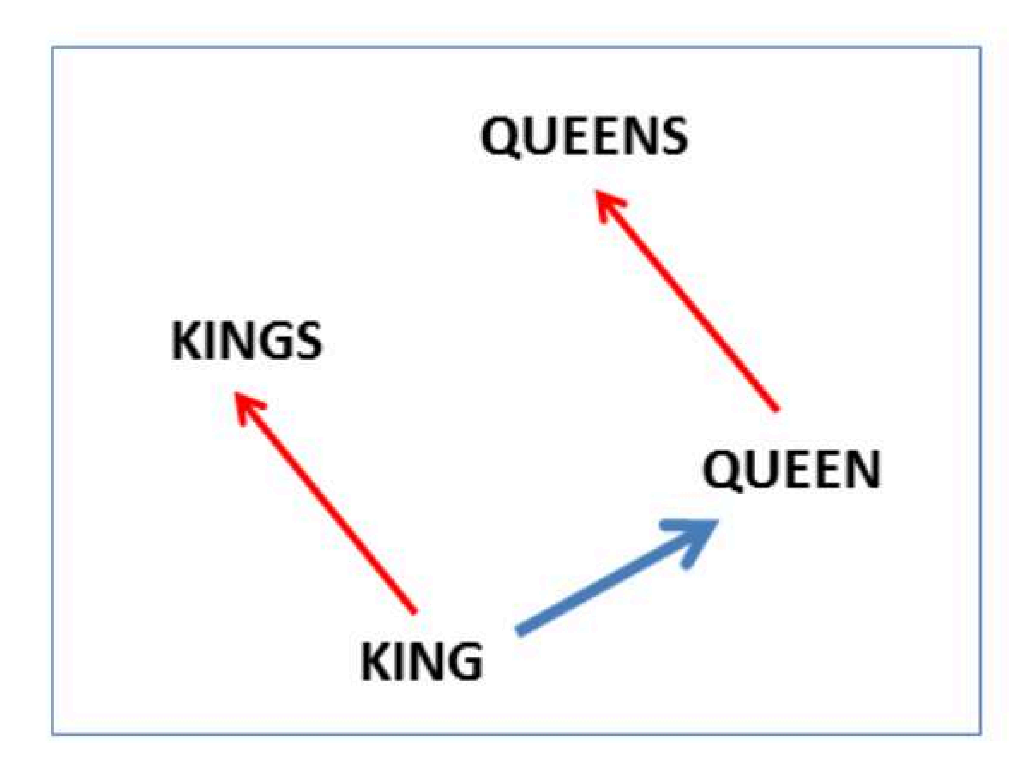


"El rey y la reina de España, junto con el hombre y la mujer de Francia, viajó y visitó Madrid y París en verano e invierno; gobernó y reinó desde el trono del palacio y el castillo en la ciudad, con la corona, el cetro y la autoridad del rey."

Word2Vec, uno de los modelos más conocidos para crear Word Embeddings, fue desarrollado por Google en 2013. Este modelo utiliza grandes cantidades de texto para aprender a posicionar cada palabra en el *espacio vectorial*. Por ejemplo, en este modelo, las palabras "rey" y "reina" estarán cerca en el *espacio vectorial* como vemos arriba, lo que indica que tienen significados relacionados. Además, las relaciones entre palabras pueden capturarse mediante operaciones matemáticas; por ejemplo, la operación "rey - hombre + mujer" dará un vector cercano a "reina".

Estos *embeddings* son útiles en una variedad de aplicaciones de procesamiento del lenguaje natural (NLP), como la traducción automática, la clasificación de textos y el análisis de sentimientos. Al representar las palabras de esta manera, las computadoras pueden realizar tareas de lenguaje con mayor precisión y entender mejor el contexto y las relaciones entre palabras, mejorando significativamente el rendimiento de las aplicaciones de NLP.

En la siguiente ilustración se refleja el concepto de que en el espacio vectorial, existiría la misma distancia entre la palabra "Rey" y "Reyes" que entre "Reina" y "Reinas".



La razón por la que los *word embeddings* son tan útiles en aplicaciones de procesamiento del lenguaje natural (NLP) es que permiten a las computadoras entender no sólo las palabras individuales, sino también las conexiones y relaciones entre ellas. Acá te explico cómo esto mejora diferentes aplicaciones:

1. **Traducción automática**: En la traducción, es crucial que la computadora entienda el significado de las palabras en su contexto. Con *word embeddings*, palabras con significados similares o relacionadas estarán cerca unas de otras en un espacio matemático. Esto significa que si una palabra no tiene una traducción directa, la computadora puede encontrar una palabra cercana en significado y utilizarla, mejorando la precisión de la traducción.
2. **Clasificación de textos**: Al analizar y clasificar textos, es importante que la computadora detecte temas o categorías basados en las palabras usadas. Gracias a los *word embeddings*, palabras relacionadas con un mismo tema estarán cerca en ese espacio matemático. Esto ayuda a la computadora a agrupar textos con temas similares, incluso si no usan exactamente las mismas palabras.
3. **Análisis de sentimientos**: Para determinar si un texto expresa un sentimiento positivo, negativo o neutral, la computadora debe captar matices de significado. *Word embeddings* permiten que la computadora entienda cómo las palabras están relacionadas en términos de sentimientos. Por ejemplo, palabras como "feliz" y "alegre" estarán cerca en el espacio de embeddings, lo que ayuda a la computadora a reconocer que ambas palabras expresan un sentimiento positivo.

Word Embeddings: La Evolución del Procesamiento del Lenguaje Natural

El Inicio de los Word Embeddings

Los word embeddings marcaron un hito en el campo del Procesamiento del Lenguaje Natural (NLP), al permitir una comprensión más profunda del significado y las relaciones entre las palabras, superando los enfoques tradicionales.

Comprendiendo el Significado y las Relaciones

Los word embeddings capturan el contexto y la semántica de las palabras, posibilitando la realización de operaciones como la identificación de analogías, abriendo así nuevas posibilidades en el NLP.

Nuevos Niveles de Precisión y Utilidad

Los word embeddings se han convertido en un componente esencial de los sistemas modernos de NLP, impulsando avances significativos en la comprensión del lenguaje natural.

Los Word Embeddings son entonces en mi opinión de una importancia espectacular para el progreso reciente del procesamiento del lenguaje natural (NLP) porque consiguieron transformar las palabras en representaciones matemáticas que capturan sus significados y relaciones contextuales. Como ya vimos, permite a los modelos de inteligencia artificial entender mejor el contexto y las similitudes entre palabras, mejorando significativamente tareas como la traducción, el análisis de sentimientos, la clasificación de textos y en general, son uno de los componentes principales en el desarrollo de los nuevos modelos de lenguaje. Esta es la era de los LLM (*Large Language Models*), con su asombrosa capacidad para generar texto que parece escrito por humanos, y que ha abierto las puertas a un universo de posibilidades, desde la creación de contenido hasta la programación y más allá.

Con esto llegamos al final de esta mini sección de *indagando en mis recuerdos*, pero ¿por qué te remarco estos dos acontecimientos? (BERT y Word Embeddings) Bueno… por dos motivos. El primero es que **sucedieron hace "nada"**. Es decir, estamos hablando de sucesos de alrededor de diez años atrás (año más, año menos), aunque **su aplicación de forma efectiva es de hace no más de cinco años**. El segundo motivo es porque incluso esos dos acontecimientos son incomparables con el nivel de progreso y el **avance tecnológico que**

estamos viviendo <u>ahora mismo</u>. Y he aquí la clave de por qué el *Primer Paso* de ENTRAR AHORA es tan importante y es por que:

Esto recién empezó, pero el crecimiento será exponencial.

Los avances son tan extremadamente rápidos que en el tiempo que haya transcurrido entre que escribo estas líneas y que vos las leas, los modelos, empresas y servicios alrededor de la IA pueden haber cambiado rotundamente[18].

Me sucedió hace poco, entre los años 2021 y 2022 impartí cursos de procesamiento de lenguaje natural para algunos de mis clientes, y si bien ya se hablaba de GPT y GPT 2, lo normal era llegar a enseñar solamente hasta BERT, que si bien como ya vimos, fue uno de los primeros modelos basados en la innovadora arquitectura de Transformers, no estaba en absoluto en un nivel cercano a lo que consiguió GPT especialmente con GPT 3 hacia fines de 2022. Por lo tanto el *hype* actual de la IA no comenzó sino hasta principios de 2023 (apenas el año anterior a que yo comenzara a escribir estas líneas).

He aquí el capitalismo, con todos sus problemas, pero también con todo su potencial de desatar una revolución productiva avasallante.

Empresas líderes y startups de todo el mundo están ahora en una carrera frenética por desarrollar y perfeccionar los mejores modelos de lenguaje, cada uno buscando superar los límites de lo posible dentro de la IA generativa. Otras empresas se focalizan en cómo llevarlos a la práctica y cómo generar verdadero impacto con ellos.

Es casi literalmente como una nueva fuente de energía que si se utiliza correctamente puede conseguir lo imposible.



[18] Por ejemplo, en la que es posible sea mi última edición antes de publicar este libro, hoy mismo 20 de Diciembre de 2024, OpenAI acaba de anunciar su nuevo modelo o3 y pareciera que es un paso muy directo al logro de la *inteligencia artificial general (AGI).*

Por último, mencionar que:

Es el momento de la IA Generativa.

Generativa porque está catapultada por modelos que se entrenan para *generar* contenido imitando los datos de entrenamiento[19]. *Datos* que son extremadamente abundantes (todo el texto de la web, todos los videos online, por poner dos ejemplos) que combinados con la impresionante capacidad de cómputo (mega infraestructuras de hardware que parecen completas ciudades[20]) permiten cosas que antes eran literalmente imposibles.

Estos modelos de lenguaje son mucho más que simples herramientas para automatizar tareas; se han convertido en compañeros de colaboración que amplían nuestra capacidad para generar ideas, comunicarnos y entender el mundo. Estamos en un punto de inflexión en el que la IA está transformando fundamentalmente la forma en que vivimos, trabajamos y creamos, ofreciéndonos una visión de un futuro que apenas estamos comenzando a imaginar.

Entonces ya estamos de acuerdo en que ***ahora*** es el mejor momento para que te metas en el mundo de la inteligencia artificial, pero…

¿Cómo empiezo?

¿Cuál es la mejor forma de abordar semejante desafío?

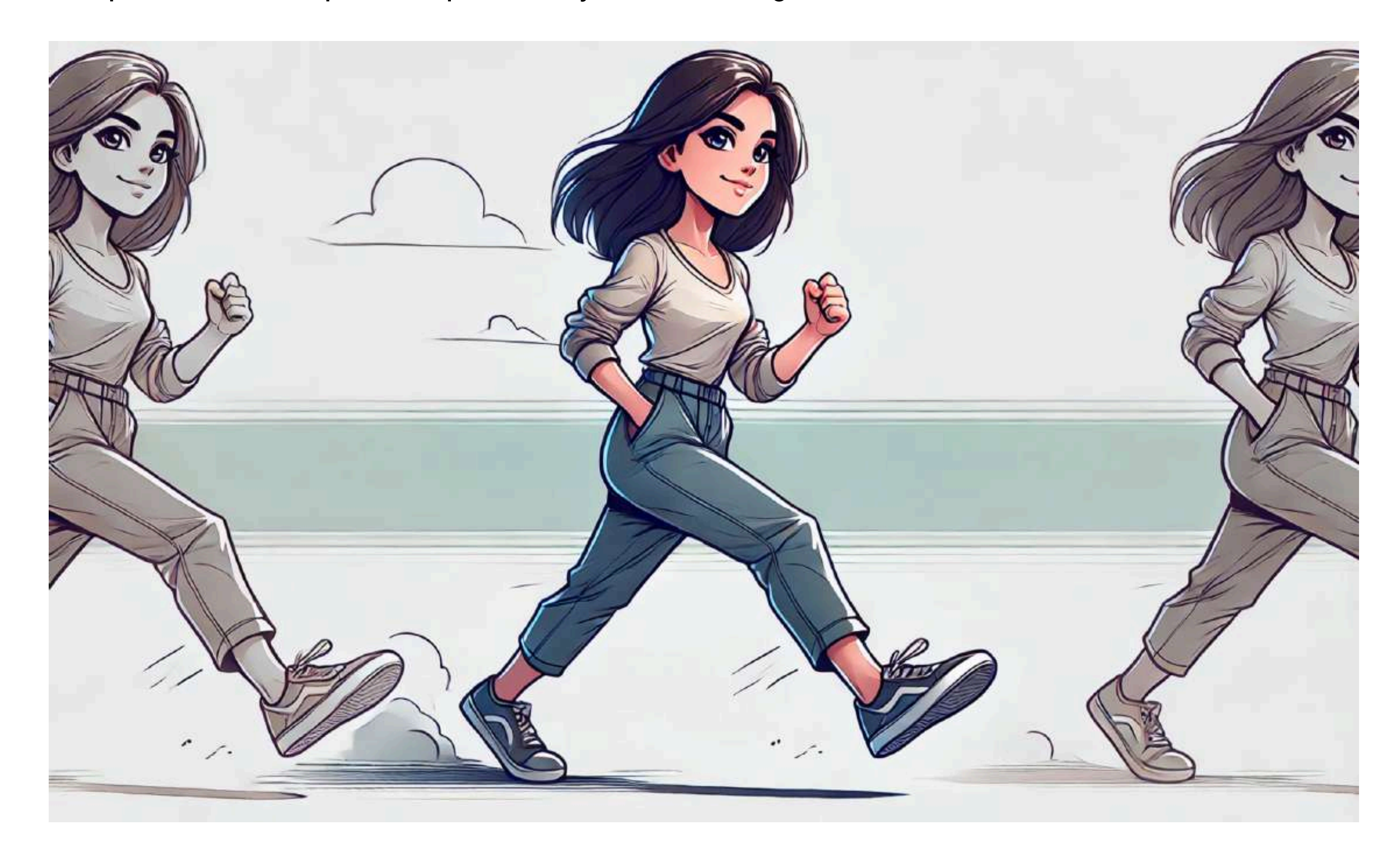
Yo te propongo dar un *Segundo Paso* muy sencillo, pero uno que nos va a tomar tal vez el 80% del libro por lo importante que es en sí mismo.

Este paso se trata simplemente de ayudarte a entender los conceptos *básicos* de la IA de forma clara y sencilla, y al mismo tiempo ofrecerte un material de consulta al que puedas acudir por cualquier desafío al que te enfrentes de ahora en más relacionado con la inteligencia artificial. Mi objetivo es que tengas un primer contacto con prácticamente todos los temas que componen la disciplina de la inteligencia artificial sin entrar en un terreno técnico. Con el paso del tiempo, con más formación y dedicación, cualquiera de estos conceptos vas a poder dominarlo y vas a poder dejar de tener sólo un conocimiento básico

[19] Existe un debate activo sobre los derechos de autor del contenido utilizado para entrenar estos modelos, tema que excede el alcance de este libro.
[20] El entrenamiento y uso de estos modelos requiere un consumo energético significativo, existiendo un debate sobre su impacto ambiental, tema que también excede el alcance de este libro.

para tener uno avanzado, pero conseguir eso será posible sólo cuando sigas el *Tercer Paso*, qué lógicamente vendrá más adelante.

Así que ahora acompañame por favor y demos el *Segundo Paso*…



Segundo Paso: aprender los BÁSICOS de la IA

Subcampos de la Inteligencia Artificial

1 Aprendizaje Automático

Las máquinas aprenden de datos y patrones para mejorar su rendimiento a través de técnicas como aprendizaje supervisado, no supervisado y por refuerzo.

2 Procesamiento de Lenguaje Natural

La IA puede comprender e interpretar el lenguaje humano, permitiendo una interacción más natural.

3 Visión por Computadora

La IA puede analizar e interpretar imágenes y videos, como reconocimiento facial y detección de objetos.

4 Robótica

La IA en robots les permite interactuar de manera autónoma con el entorno físico.

¿Qué es exactamente la Inteligencia Artificial?

La Naturaleza de la Inteligencia

La inteligencia es un concepto complejo que abarca múltiples aspectos del funcionamiento cognitivo humano. Cuando consideramos qué hace inteligente a una persona, podemos identificar varios componentes clave:

1 Base de Conocimiento

Poseer una amplia base de conocimiento es fundamental para la inteligencia. Esto implica tener información y comprensión sobre diversos temas y áreas.

2 Aplicación del Conocimiento

Saber utilizar herramientas para aplicar ese conocimiento en la resolución de problemas es otro aspecto crucial de la inteligencia. No solo se trata de tener información, sino de saber cómo usarla de manera efectiva.

3 Aprendizaje y Mejora Continua

Mejorar con la experiencia y cada vez equivocarse menos es una característica clave de la inteligencia. Esto implica la capacidad de aprender de los errores y adaptar el comportamiento en consecuencia.

4 Memoria y Contextualización

Tener una buena memoria y saber reconocerse en contextos y entornos es esencial para la inteligencia. Esto permite a una persona aplicar conocimientos previos a nuevas situaciones de manera efectiva.

Además de estos componentes, un aspecto crucial de la inteligencia es la capacidad de relacionar y asociar conocimiento, experiencias, herramientas, contextos y entornos para reaccionar a situaciones nuevas. Esta habilidad de síntesis y adaptación es lo que realmente distingue a una mente inteligente, permitiéndole navegar eficazmente por situaciones desconocidas y resolver problemas complejos.

Comencemos el segundo paso por lo más básico de lo básico: la definición de *Inteligencia Artificial* (que podemos conseguir con ChatGPT por ejemplo). Ya que estamos hablando de IA, es importante en este punto que la definamos correctamente:

> La Inteligencia Artificial (IA) es el campo de la informática que se enfoca en la creación de sistemas capaces de realizar tareas que normalmente requieren de la inteligencia humana. Estas tareas incluyen, pero no se limitan a, el aprendizaje, el razonamiento, la resolución de problemas, la

comprensión del lenguaje natural, la percepción visual y la toma de decisiones.

También consideremos algunos de los componentes claves de la IA, aunque en este capítulo vamos a ver en detalle todos los que yo personalmente considero los más importantes (*spoiler alert*: son trece)

Componentes clave de la IA

1. **Aprendizaje Automático (*Machine Learning*)**:
    a. **Supervisado**: El modelo aprende de un conjunto de datos etiquetados, donde las respuestas correctas están proporcionadas.
    b. **No supervisado**: El modelo encuentra patrones y relaciones en datos no etiquetados.
    c. **Aprendizaje por refuerzo (*Reinforcement Learning*)**: El sistema aprende a tomar decisiones mediante la obtención de recompensas o penalizaciones.
2. **Procesamiento de Lenguaje Natural (*Natural Language Processing*, NLP)**: Capacidad de las máquinas para entender, interpretar y generar lenguaje humano de manera útil y significativa.
3. **Visión por Computadora (*Computer Vision*)**: Habilidad de las máquinas para interpretar y entender el contenido visual del mundo, como imágenes y videos.
4. **Robótica**: Integración de IA en robots para permitirles interactuar con el entorno físico de manera autónoma.
5. **Sistemas Expertos**: Programas que emulan la capacidad de toma de decisiones de un experto humano en un dominio específico.

Planteada ésta definición y algunos primeros conceptos clave, quiero pasar a explicarte lo que **yo** considero importante cuando pienso en el concepto de *Inteligencia Artificial*.

Empecemos por la parte más obvia:

- **Artificial**. Algo que no es natural, es decir, algo creado o producido por seres humanos utilizando técnicas y tecnología.

Y la parte no tan obvia:

- **Inteligencia**. Cuando pienso en qué hace más o menos inteligente[21] a una persona, en general considero las siguientes características:
    - La *base de conocimiento* que posee
    - Su capacidad de utilizar *herramientas* para aplicar ese conocimiento en la resolución de problemas
    - Su capacidad de mejorar con la *experiencia* y cada vez equivocarse menos en la resolución de dichos problemas

[21] Dejemos de lado por ahora la realidad de que existen diferentes tipos de inteligencia que conectan más con las emociones humanas.

- Su capacidad de *memorizar*
- Su capacidad de *relacionar* y *asociar* conocimiento, experiencias, herramientas, contextos y entornos para reaccionar a situaciones nuevas.

Lógicamente la inteligencia total de un individuo o sistema será una combinación de esos puntos anteriores (y otros que me puedan faltar) y sus diferentes grados. Esto nos es útil porque el camino de la inteligencia artificial hasta el día de hoy, es un camino lleno de progreso que va hacia un sistema completamente inteligente (de grado máximo), pero que ha comenzado desde los grados más bajos. Es una historia marcada por avances, retos superados y una evolución constante que ha redefinido lo que es posible. Desde sus humildes comienzos en la década de 1950, cuando la idea de "máquinas pensantes" pasó de la ciencia ficción a la academia, la inteligencia artificial viene siguiendo un camino lleno de innovaciones y descubrimientos.

Hasta hace poco era difícil pensar en una inteligencia artificial que cumpliera con no solo varios sino con sólo alguno de los puntos anteriores (base de conocimiento, utilización de herramientas, entendimiento del contexto, memoria, capacidad de relacionar). Por otro lado, hoy en día con la irrupción de los LLM (ChatGPT un ejemplo de ellos) y el mejor aprovechamiento de ellos, la IA basada en *Agentes* (te lo voy a explicar más adelante) ya prácticamente podemos decir que la IA consigue un grado alto de casi **todas** las características anteriores y nos lleva a pensar que estamos en la antesala de la ***Inteligencia Artificial General***.

La Inteligencia Artificial General (IAG) se refiere a un tipo de inteligencia artificial que tiene la capacidad de realizar cualquier tarea intelectual de manera similar o superior a la que un ser humano la puede hacer. A diferencia de la IA específica o *estrecha (o débil)*, que está diseñada para resolver problemas concretos (como la generación de texto, el reconocimiento de rostros o la traducción de idiomas), la IAG sería capaz de razonar, aprender y aplicar conocimiento en múltiples dominios de manera autónoma y adaptable. Su objetivo es replicar la flexibilidad cognitiva humana, permitiendo que la *máquina* entienda y resuelva problemas nuevos sin intervención humana.

Los casos de uso de *agentes de IA* en la actualidad que se construyen gracias a las nuevas capacidades de los LLM casi todos poseen:

1. Una **base de conocimientos** en forma de base de datos documental vectorizada (es decir muchísimos documentos transformados a representaciones numéricas). Adicionalmente se podría considerar como base de conocimiento a todo el texto con el que se ha entrenado al propio modelo (por ejemplo todo el contenido de Wikipedia).
2. **Herramientas** disponibles que el *agente* puede utilizar para resolver problemas, en la forma de *funciones* escritas en lenguajes de programación (Python por ejemplo) o el acceso a APIs

(forma de utilizar herramientas de software desarrolladas por terceros). Por ejemplo, una función que le permita al *agente* hacer inversiones en activos financieros.

3. Capacidad de mejorar con la **experiencia** a través de técnicas como el aprendizaje por refuerzo (lo veremos más adelante).

4. Capacidad de **memoria** cada vez mayor, medida generalmente como la cantidad de *tokens* (similar a *palabras*) que son capaces de mantener en su *contexto* (actualmente los mejores modelos pueden tener en su *memoria* incluso más texto que la colección completa de libros de Harry Potter, por poner un ejemplo).

5. Capacidad de **entender el entorno** y *escuchar* señales que le permitan reaccionar y adaptarse. Por ejemplo, activarse ante un evento económico importante anunciado.



¿Llegarán estos sistemas a la **IAG**? Algunos expertos creen que alcanzar la IAG es cuestión de tiempo y avances en las técnicas actuales de IA, como el *deep learning* y los modelos generativos. Otros sostienen que las tecnologías actuales están lejos de poder replicar la complejidad y flexibilidad de la inteligencia humana, argumentando que se necesitan nuevos enfoques o paradigmas para avanzar.

Lo cierto es que nunca lo tuvimos tan cerca y en mi opinión, veremos un mundo mejor y más próspero gracias a la IA. Por eso quiero ayudarte a que te prepares y aprendas todos los conceptos básicos y que este sea el comienzo de tu propio camino con la inteligencia artificial.

La Inteligencia Artificial en acción: del problema a la solución

Pero antes, para entender verdaderamente el poder y el propósito de la Inteligencia Artificial, nos sería útil pensar cómo suele abordar los problemas complejos del mundo real y de qué nos sirve verdaderamente la IA. Como señalan los profesores Dorsa Sadigh y Percy Liang[22] en uno de mis cursos favoritos de IA que se encuentra en Stanford Online (CS221), nos encontramos constantemente con situaciones llenas de "messiness and uncertainty" (incertidumbre y complejidad): desde navegar por una ciudad congestionada hasta diagnosticar enfermedades o tomar decisiones financieras. Como explica la profesora Sadigh, "estamos interesados en resolver problemas realistas y complejos, que tienen mucho desorden e incertidumbre".

La magia de la IA reside entonces en su capacidad para transformar estos problemas que se nos presentan desordenados y complejos en algo que una computadora o un sistema pueda procesar y resolver. Este proceso ocurre en tres etapas fundamentales:

- **Modelado**. Primero, la IA modela el problema, creando una representación simplificada pero útil de la realidad. Imaginá que querés que un sistema de IA ayude a gestionar el tráfico en tu ciudad. No puede procesar toda la complejidad de la vida urbana, pero puede representar las calles como líneas conectadas y las intersecciones como puntos, creando un mapa que captura lo esencial del problema.
- **Inferencia**. Segundo, el sistema utiliza este modelo para razonar y hacer inferencias. Una vez que tenemos un "mapa" del problema, podemos comenzar a buscar soluciones: ¿cuál es la mejor ruta para llegar al trabajo? ¿Cómo podemos sincronizar los semáforos para reducir la congestión? La IA puede procesar miles de posibilidades en segundos, encontrando patrones y soluciones que podrían no ser evidentes para el ojo humano.
- **Aprendizaje**. Tercero, y quizás lo mejor que nos ofrece la IA es que aprende y mejora con el tiempo. A diferencia de un software tradicional que sigue reglas fijas, los sistemas modernos de IA aprenden de los datos y la experiencia. Observan patrones

[22] Este texto se basa parcialmente en el curso CS221: Artificial Intelligence: Principles and Techniques (Otoño 2021) de la Universidad de Stanford, impartido por los profesores Dorsa Sadigh y Percy Liang. El curso presenta una introducción fundamental a la IA a través de tres pilares: modelado, inferencia y aprendizaje. La profesora Sadigh es Profesora Asistente de Ciencias de la Computación, especializada en robótica e IA, particularmente en la interacción entre agentes robóticos/IA con humanos. El profesor Liang, experto en aprendizaje automático y procesamiento del lenguaje natural, se centra en hacer los sistemas más robustos y confiables, con especial interés en modelos fundamentales. https://www.youtube.com/watch?v=ZiwogMtbjr4&list=PLoROMvodv4rOca_Ovz1DvdtWuz8BfSWL2&ab_channel=StanfordOnline

en el tráfico real, aprenden de las decisiones exitosas y fallidas, y ajustan constantemente sus modelos para ser más precisos y útiles.

Esta capacidad de modelar, razonar y aprender es lo que hace que la IA sea tan poderosa. No se trata solo de seguir instrucciones programadas, sino de desarrollar una comprensión cada vez más sofisticada del mundo y cómo funciona. Es por eso que la IA puede ayudarnos en tareas tan diversas como:

- Detectar enfermedades en imágenes médicas con precisión comparable a la de los mejores especialistas
- Predecir patrones climáticos y ayudar a prepararnos para eventos meteorológicos extremos
- Optimizar el consumo de energía en edificios y ciudades enteras
- Identificar fraudes financieros en tiempo real
- Personalizar experiencias educativas para cada estudiante

La verdadera revolución de la IA moderna es que estas capacidades están creciendo y exponencialmente. Los sistemas actuales pueden combinar el procesamiento del lenguaje con el análisis visual, el razonamiento lógico con el aprendizaje experiencial, creando herramientas cada vez más versátiles y poderosas.

A medida que avancemos en este libro, vamos a explorar cada uno de los componentes que yo considero son los más básicos y que hacen posible esta *magia*: desde las estructuras lógicas básicas hasta los sistemas de aprendizaje más avanzados. Vamos a ver cómo cada pieza encaja en el rompecabezas más grande de la inteligencia artificial y, lo más importante, cómo podemos aprovechar estas herramientas para resolver problemas reales y crear un impacto positivo en el mundo. Ahora sí, empecemos!

Conceptos básicos de la Inteligencia Artificial

Como te lo prometí, el objetivo de este libro no es que salgas siendo una persona experta en Inteligencia Artificial, sino que puedas entender qué tenés qué hacer para convertirte en una si lo quisieras. Me interesa que tengas a mano un recurso para rápidamente poder consultar y que te ofrezca un panorama general de todos los conceptos importantes. Por eso ahora vamos a meternos con los que yo considero los conceptos *básicos* de esta disciplina.[23]

1. 📐 **Lógica Proposicional**
2. 🔎 **Lógica de Primer Orden**
3. 🎯 **Aprendizaje Supervisado (Machine Learning)**

[23] Como en el resto del libro, todos los textos en recuadros han sido trabajados con modelos LLM diversos y preparados de la forma más didáctica posible y adaptada al objetivo primordial de este libro.

4. 🔍 **Aprendizaje No Supervisado (Machine Learning)**
5. 🕹️ **Aprendizaje por Refuerzo (Reinforcement Learning)**
6. 🧠 **Redes Neuronales / Deep Learning**
7. 🗣️ **Procesamiento de Lenguaje Natural (NLP)**
8. 👁️ **Visión por Computadora (Computer Vision)**
9. 🧑‍💻 **Sistemas Expertos**
10. 🔗 **Redes Bayesianas**
11. 🚀 **Transformers y Modelos de Lenguaje de Gran Escala (LLMs)**
12. 🥷 **Agentes IA (Agentive AI)**
13. 🦾 **Robótica**

Estos temas los considero *los básicos* de la inteligencia artificial porque abarcan los fundamentos teóricos y prácticos que sustentan las diferentes áreas y sus aplicaciones. A continuación, te explico la importancia de cada uno antes de que los veamos en profundidad:

1. 📐 **Lógica Proposicional**: Es uno de los pilares más antiguos de la IA. Proporciona las bases matemáticas para representar el razonamiento lógico. Es fundamental en la IA simbólica, donde se busca representar el conocimiento a través de reglas lógicas.

2. 🔎 **Lógica de Primer Orden**: Extiende la lógica proposicional permitiendo la representación de relaciones más complejas y la cuantificación de variables. Es esencial para sistemas que necesitan un razonamiento más profundo y la manipulación de conocimiento abstracto, como los sistemas expertos.

3. 🎯 **Aprendizaje Supervisado (Supervised *Machine Learning*)**: Es una técnica clave del *machine learning* donde un modelo aprende de ejemplos etiquetados para hacer predicciones o clasificaciones. Es ampliamente utilizado en aplicaciones como la predicción de ventas, reconocimiento de voz y procesamiento de imágenes.

4. 🔍 **Aprendizaje No Supervisado (*Unsupervised Machine Learning*)**: Se centra en encontrar patrones o estructuras en datos no etiquetados. Es básico en la IA porque permite descubrir relaciones ocultas, como en la segmentación de clientes o la detección de anomalías.

5. 🕹️ **Aprendizaje por Refuerzo (*Reinforcement Learning*)**: Este paradigma permite que agentes aprendan a tomar decisiones mediante la interacción con su entorno y la maximización de recompensas. Es crucial en el desarrollo de sistemas autónomos y la robótica.

6. 🧠 **Redes Neuronales / Deep Learning**: Son inspiradas por el cerebro humano y permiten a los sistemas aprender representaciones complejas de datos. El *Deep Learning* ha impulsado grandes avances en tareas como visión por computadora, reconocimiento de voz y procesamiento de lenguaje natural.

7. 🗣️ **Procesamiento de Lenguaje Natural (NLP)**: Se centra en la comprensión e interpretación del lenguaje humano por parte de las máquinas. Es fundamental para chatbots, traducción automática y análisis de sentimientos, facilitando la interacción entre humanos y máquinas.

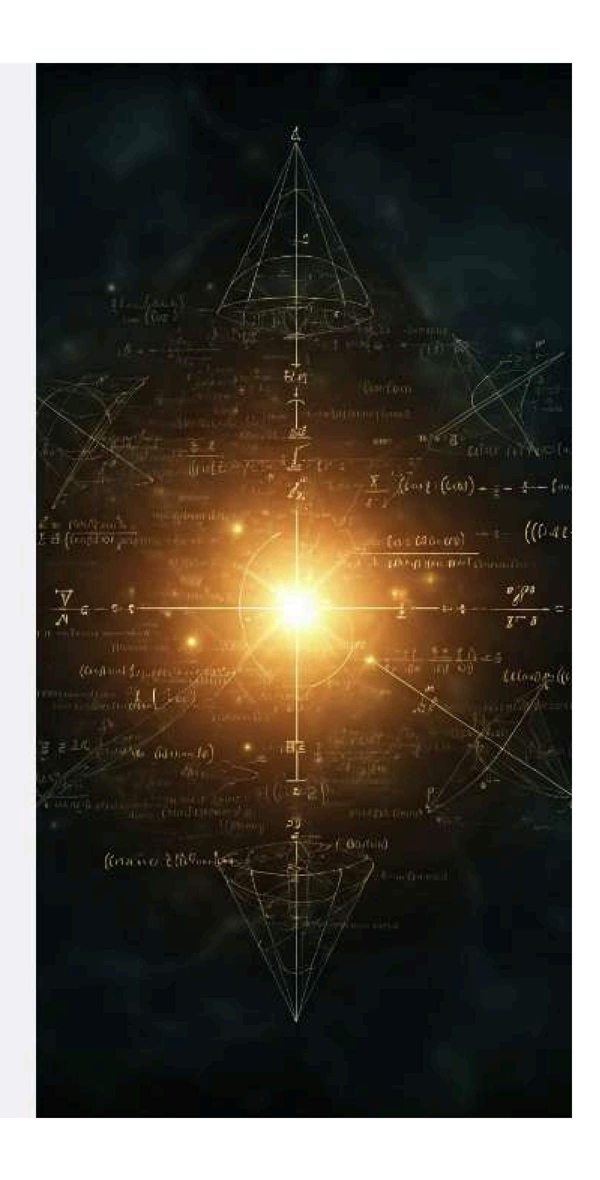
8. 👁️ **Visión por Computadora (Computer Vision)**: Permite a las máquinas interpretar y procesar imágenes del mundo real. Es básico en aplicaciones como conducción autónoma, diagnóstico médico por imágenes y reconocimiento facial.
9. 🧑‍💻 **Sistemas Expertos**: Son programas que emulan el juicio de un experto humano en un dominio específico. Tienen aplicaciones en medicina, derecho, y diagnóstico técnico, donde es necesario capturar y utilizar conocimiento especializado.
10. 🔗 **Redes Bayesianas**: Son modelos probabilísticos que permiten el razonamiento bajo incertidumbre. Son esenciales en problemas donde se necesita tomar decisiones o hacer inferencias con datos incompletos o ruidosos, como el diagnóstico médico.
11. 🚀 **Transformers y Modelos de Lenguaje de Gran Escala (LLMs)**: Estas arquitecturas han revolucionado el procesamiento de lenguaje natural y la generación de texto. Son la base de los modelos modernos como GPT, Llama y Claude, que permiten la creación de sistemas avanzados de lenguaje.
12. 🥷 **Agentes IA (Agentive AI)**: La Inteligencia Artificial basada en agentes se refiere a sistemas autónomos que perciben su entorno, toman decisiones y ejecutan acciones para alcanzar objetivos específicos, adaptándose y aprendiendo de sus experiencias.
13. 🦾 **Robótica**: La robótica es fundamental en la IA porque integra diversas áreas, como el aprendizaje automático, la visión por computadora, y la planificación. Permite la interacción física con el mundo, haciendo posible la automatización de tareas en el mundo real.

Con esta pequeña introducción finalizada, ahora sí vamos a ver cada uno de estos conceptos básicos detalladamente.

01. 📐 Lógica Proposicional

Lógica Proposicional: Fundamentos del Razonamiento

La lógica proposicional es un sistema formal que utiliza proposiciones, que son enunciados que pueden ser verdaderos o falsos, y conectores lógicos para representar y manipular conocimiento. Es un pilar fundamental de la inteligencia artificial y la ciencia de la computación, permitiendo la creación de sistemas de razonamiento automático.



Comenzamos nuestro recorrido por los *básicos* de la IA con uno de los conceptos que normalmente es omitido pero que yo considero que es fundamental. La lógica proposicional es clave en la inteligencia artificial porque permite modelar y razonar de manera precisa sobre problemas complejos. Por ejemplo, en un sistema de IA, si se tiene la proposición "Está lloviendo" (P) y "Voy a llevar un paraguas" (Q), se puede usar la implicación lógica ($P \rightarrow Q$) para que la *máquina* decida automáticamente llevar un paraguas cuando detecta lluvia. También es fundamental para verificar que un algoritmo de un automóvil autónomo frene correctamente (P) si detecta un obstáculo (Q), usando reglas lógicas para asegurar la seguridad.

La lógica proposicional sienta las bases para que las computadoras puedan "pensar" de manera sistemática, evaluando si las afirmaciones son verdaderas o falsas en diferentes situaciones. Esta capacidad de razonamiento lógico es esencial para que las máquinas puedan tomar decisiones basadas en reglas claras y precisas. Así como los humanos usamos el razonamiento lógico para decidir si llevamos un paraguas cuando vemos nubes

oscuras, las máquinas necesitan un sistema similar pero más riguroso para procesar información y tomar decisiones. Vamos a ver más adelante cómo esta base lógica es fundamental en técnicas avanzadas como los árboles de decisión, donde cada decisión se basa en una serie de proposiciones lógicas, o en las redes neuronales, donde los operadores lógicos toman formas más sofisticadas para permitir el aprendizaje automático.



La lógica proposicional es uno de los fundamentos de la inteligencia artificial y de la ciencia de la computación. Es una rama de la lógica que se ocupa de las proposiciones y sus relaciones a través de conectores lógicos. En esencia, se enfoca en la manipulación y análisis de proposiciones verdaderas o falsas.

Conceptos Clave

- **Proposición**: Una proposición es una declaración que puede ser verdadera o falsa. Por ejemplo, "La tierra es plana" es una proposición (falsa).
- **Conectores Lógicos**: Son operadores que combinan proposiciones para formar proposiciones más complejas. Los principales conectores lógicos son:
    - Negación (¬): Invierte el valor de verdad de una proposición. Si P es verdadero, ¬P es falso.
    - Conjunción (∧): P ∧ Q es verdadero si y sólo si tanto P como Q son verdaderos.
    - Disyunción (∨): P ∨ Q es verdadero si al menos una de las proposiciones es verdadera.
    - Implicación (→): P → Q es verdadero excepto cuando P es verdadero y Q es falso.
    - Doble implicación (↔): P ↔ Q es verdadero si P y Q tienen el mismo valor de verdad.

Proposiciones

Una proposición es una declaración que puede ser verdadera o falsa.

Ejemplos:

- P: "La Tierra es redonda." (Verdadera)
- Q: "Los gatos pueden volar." (Falsa)

Conectores Lógicos

Negación (¬): Invierte el valor de verdad de una proposición.

Ejemplo: ¬P = "La Tierra no es redonda."

Conjunción (∧): Verdadera si ambas proposiciones son verdaderas.

Ejemplo: P ∧ Q = "La Tierra es redonda y los gatos pueden volar."

Disyunción (∨): Verdadera si al menos una proposición es verdadera.

Ejemplo: P ∨ Q = "La Tierra es redonda o los gatos pueden volar."

Implicación (→): Falsa solo si el antecedente es verdadero y el consecuente falso.

Ejemplo: P → Q = "Si la Tierra es redonda, entonces los gatos pueden volar."

Doble implicación (↔): Verdadera si ambas proposiciones tienen el mismo valor de verdad.

Ejemplo: P ↔ Q = "La Tierra es redonda si y solo si los gatos pueden volar."

Ejemplo de Lógica Proposicional

Supongamos que tenemos las siguientes proposiciones:

- P: "Está lloviendo."
- Q: "Voy a llevar un paraguas."

Podemos formar proposiciones más complejas usando conectores lógicos:

- P ∧ Q: "Está lloviendo y voy a llevar un paraguas."
- P ∨ Q: "Está lloviendo o voy a llevar un paraguas."
- ¬P: "No está lloviendo."

- P → Q: "Si está lloviendo, entonces voy a llevar un paraguas."

Estas expresiones se pueden evaluar en función de la verdad de P y Q.

Uso de la Lógica Proposicional en Inteligencia Artificial

La lógica proposicional es fundamental en muchas aplicaciones de IA. Acá hay algunos ejemplos de cómo se utiliza:

1. Sistemas Expertos

Los sistemas expertos utilizan lógica proposicional para representar el conocimiento y tomar decisiones basadas en reglas. Tenemos una sección entera dedicada a este tema en el libro, así que de momento sólo veamos un ejemplo:

> **Ejemplo:** Diagnóstico médico
>
> - P: "El paciente tiene fiebre"
> - Q: "El paciente tiene dolor de garganta"
> - R: "El paciente tiene gripe"
>
> Regla: (P ∧ Q) → R
>
> "Si el paciente tiene fiebre y dolor de garganta, entonces el paciente tiene gripe."

2. Planificación y resolución de problemas

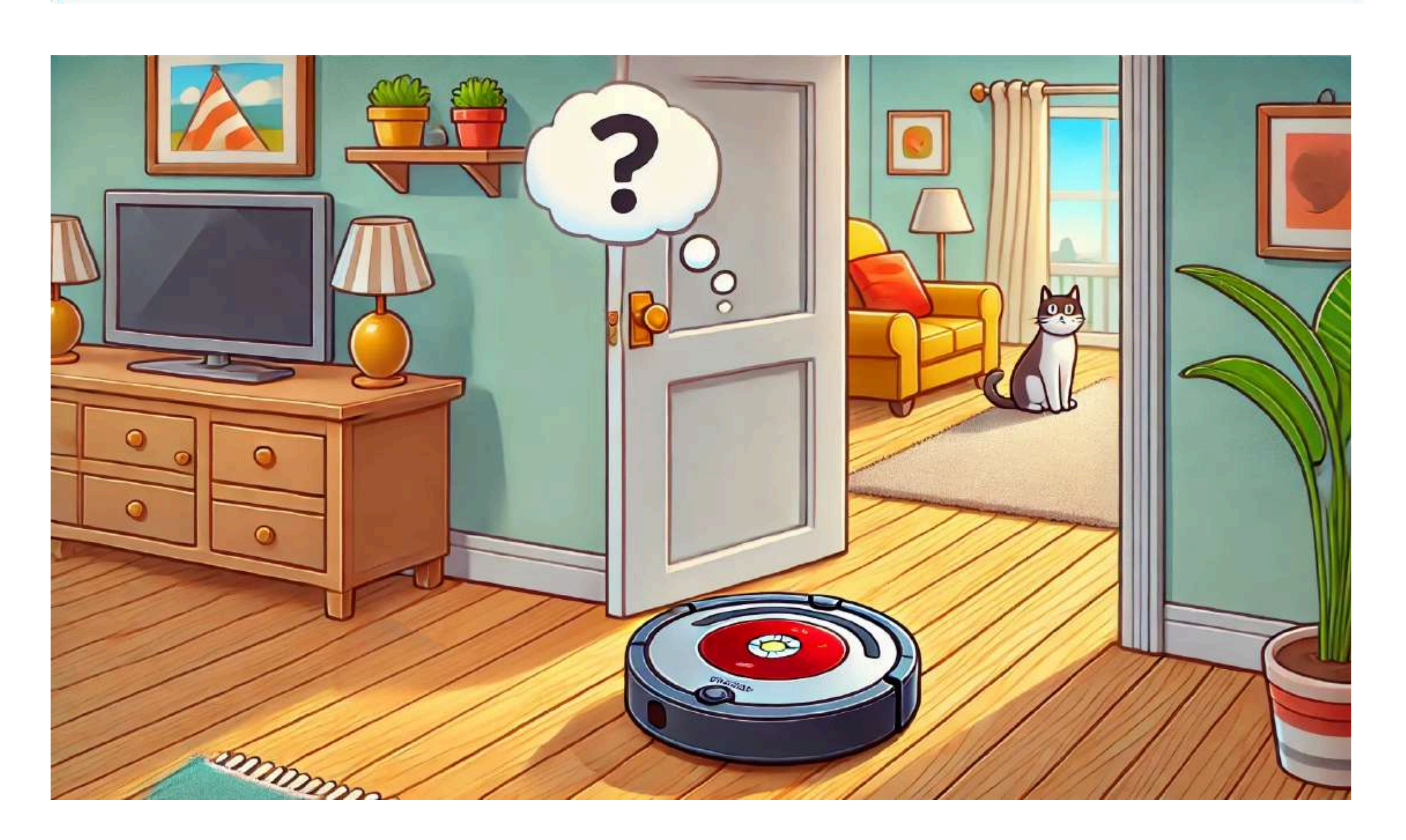
La IA utiliza lógica proposicional para representar estados y acciones en problemas de planificación. Por ejemplo:

Ejemplo: Planificación de ruta para un robot

- P: "El robot está en la habitación A"
- Q: "La puerta entre A y B está abierta"
- R: "El robot puede moverse a la habitación B"

Regla: $(P \wedge Q) \rightarrow R$

"Si el robot está en la habitación A y la puerta entre A y B está abierta, entonces el robot puede moverse a la habitación B."



3. Aprendizaje Automático (*Machine Learning*)

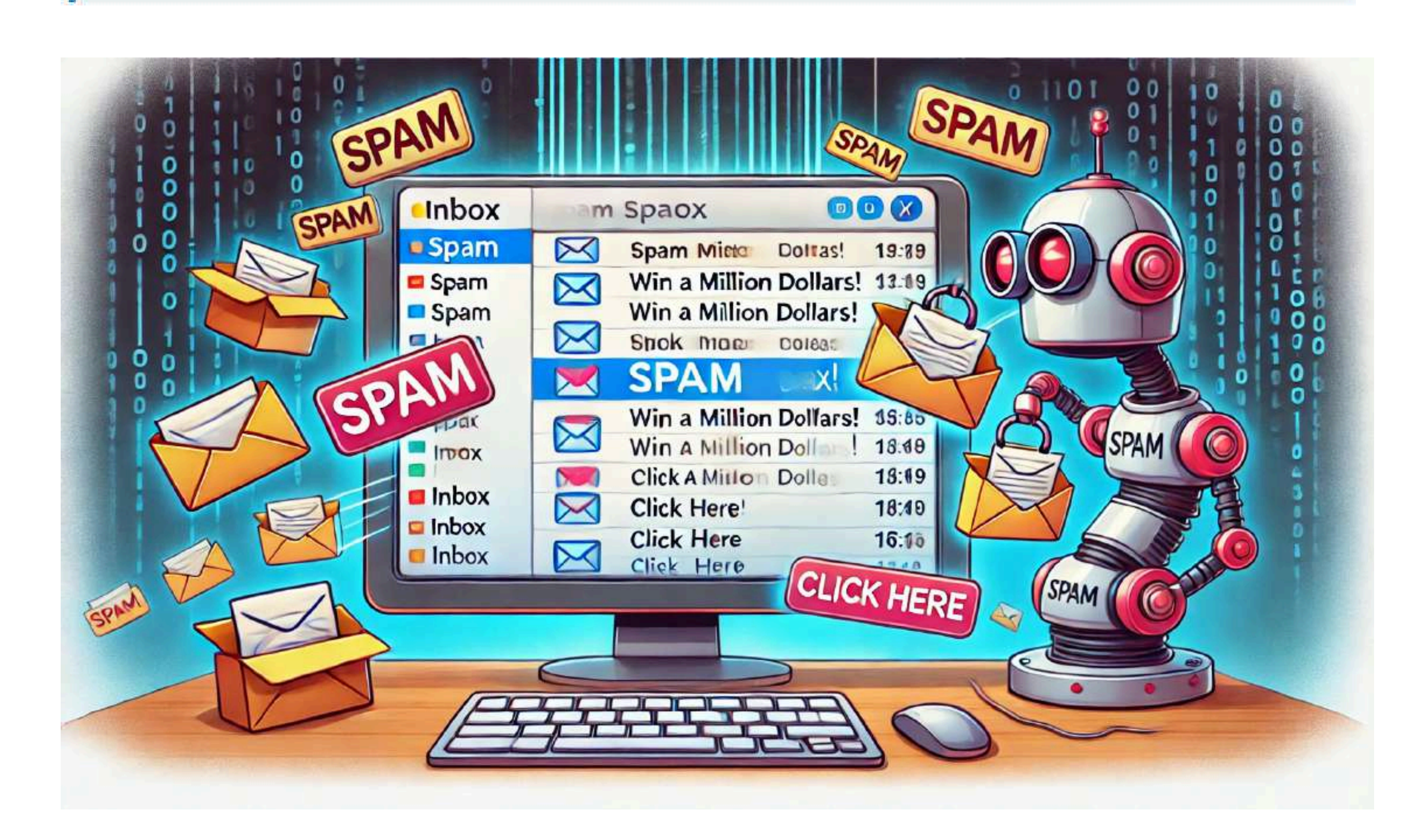
En aprendizaje automático, la lógica proposicional se utiliza para crear y evaluar características (features) en modelos de clasificación.

Ejemplo: Clasificación de correos electrónicos

- P: "El email contiene la palabra 'oferta'"
- Q: "El email proviene de un dominio desconocido"
- R: "El email es spam"

Regla: $(P \vee Q) \rightarrow R$

"Si el email contiene la palabra 'oferta' o proviene de un dominio desconocido, entonces es probable que sea spam."



La lógica proposicional, a pesar de su aparente simplicidad, tiene un impacto profundo en los sistemas tecnológicos que utilizamos diariamente. Esta rama fundamental de la lógica matemática proporciona la base para el *razonamiento binario* que sustenta la mayoría de los sistemas computacionales modernos. Por ejemplo, cada decisión que toma un dispositivo electrónico, desde un *smartphone* hasta un automóvil inteligente, se basa en la evaluación de proposiciones simples que solo pueden ser verdaderas o falsas.

Por ejemplo, cuando usamos un asistente virtual como Siri o Alexa, la lógica proposicional está actuando. Si le pedimos a nuestro asistente que nos recuerde llevar un paraguas si está lloviendo, el sistema evalúa dos proposiciones: 'está lloviendo' y 'necesita recordatorio para llevar paraguas'. Basándose en el valor de verdad de estas proposiciones, el asistente decide si debe crear un recordatorio o no. De manera similar, los sistemas de recomendación en plataformas de streaming como Netflix utilizan la lógica proposicional para evaluar múltiples condiciones sobre nuestros gustos y hábitos de visualización, determinando qué contenido sugerirnos. Incluso en algo tan cotidiano como usar una tarjeta de crédito, los sistemas de detección de fraudes usan algoritmos complejos basados en lógica proposicional para evaluar si una transacción es legítima o sospechosa. No me quiero adelantar más, porque todo esto y mucho más lo veremos en las siguientes páginas.

Desde los sistemas de aprendizaje automático hasta los algoritmos de procesamiento del lenguaje natural, la IA se basa en la evaluación y manipulación de proposiciones lógicas y es por eso que la *Lógica Proposicional* es el primer concepto básico de la IA que quiero que recuerdes siempre.

02. 🔎 Lógica de Primer Orden

Lógica de Primer Orden en Acción

Representación Formal

La lógica de primer orden permite representar el conocimiento de manera precisa y estructurada, utilizando variables, predicados y cuantificadores para capturar relaciones y propiedades entre objetos.

Razonamiento Automatizado

Los sistemas de inteligencia artificial pueden emplear la lógica de primer orden para realizar inferencias y deducciones lógicas de manera eficiente, facilitando el desarrollo de aplicaciones avanzadas.

Modelado y Optimización

Esta herramienta permite modelar problemas complejos y optimizar soluciones, lo cual es crucial en áreas como la planificación, la toma de decisiones y la resolución de problemas.

La lógica proposicional que vimos en la sección anterior, aunque muy útil, tiene sus limitaciones al tratar ideas más complejas. Acá es donde entra la *lógica de primer orden*, también llamada *lógica de predicados*. Ésta amplía la lógica proposicional agregando **variables**, **cuantificadores** y **predicados** lo cual permite hablar sobre *propiedades* de objetos y cómo se relacionan entre sí. Mientras la *lógica proposicional* solo maneja afirmaciones que son totalmente verdaderas o falsas, la *lógica de primer orden* puede tratar con sistemas más grandes y expresar ideas más generales. Por eso es crucial para el desarrollo de sistemas de IA que necesitan *razonar* sobre relaciones y propiedades entre objetos.

Junto con la lógica proposicional, estas herramientas proporcionan la base para desarrollar sistemas inteligentes capaces de razonar, planificar y tomar decisiones.
Veamos algunos conceptos claves de la lógica de primer órden.

Conceptos Clave

- **Predicados**: Son funciones que devuelven verdadero o falso. Por ejemplo, "Amigo(x, y)" puede representar "x es amigo de y".
- **Cuantificadores**:
    - **Cuantificador Existencial (∃)**: "∃x P(x)" se lee como "existe al menos un *x* tal que P(x) es verdadero".

- ○ **Cuantificador Universal (∀)**: "∀x P(x)" se lee como "para todo *x*, P(x) es verdadero".
- **Variables**: Representan elementos individuales en el dominio de discurso. Por ejemplo, en "∀x (Humano(x) → Mortal(x))", *x* es una variable.
- **Funciones**: Mapas de elementos de un dominio a otros elementos. Por ejemplo, "Padre(x)" podría representar la función que devuelve *el padre de x*. Esto es, si yo me llamo Gabriel y mi padre Mario, entonces la función Padre(x) donde x es una variable y es igual a "Gabriel" daría como resultado "Mario". Es lo mismo que decir Padre(Gabriel) = Mario. O Padre(x) = Mario donde x = Gabriel.

<u>Ejemplo de Lógica de Primer Orden</u>

> Supongamos que queremos expresar conocimiento sobre una pequeña familia:
>
> - `Humano(x)` : x es humano.
> - `Padre(y, x)` : y es el padre de x.
> - `Madre(z, x)` : z es la madre de x.
>
> Podemos escribir:
>
> - `∀x (Humano(x) → Mortal(x))` : "Todos los humanos son mortales."
> - `∃y (Padre(y, Juan))` : "Existe alguien que es el padre de Juan."
> - `∀x ∀y ((Padre(y, x) ∧ Humano(x)) → Humano(y))` : "Si y es el padre de x y x es humano, entonces y también es humano."

Estas expresiones permiten representar relaciones y propiedades más complejas y son fundamentales en el desarrollo de sistemas inteligentes que necesitan razonar sobre el mundo.

<u>Aplicaciones de la lógica de primer órden en la IA</u>

- 🤖 **Sistemas Expertos**: Utilizan *lógica proposicional* y *lógica de primer orden* para inferir conocimientos a partir de reglas y hechos conocidos (lo veremos en uno de los conceptos básicos más adelante)
- 📋 **Planificación Automatizada**: Se basa en la *lógica* para crear planes de acción que lleven a la consecución de objetivos específicos.
- 💬 **Procesamiento de Lenguaje Natural:** Emplea *lógica de primer orden* para comprender y generar lenguaje humano, permitiendo interacciones más naturales con los sistemas de IA.
- 🦾 **Robótica**: Utiliza estos conceptos para razonar sobre el entorno y tomar decisiones basadas en las percepciones y los objetivos del robot.

Desarrollemos un ejemplo sencillo para ilustrar cómo se puede utilizar la *lógica de primer orde*n (LPO) en un contexto práctico. Imaginemos que estamos trabajando en un sistema inteligente que necesita *razonar* sobre la relación entre personas y sus roles en una empresa.

Predicados:

- `Empleado(x)`: x es un empleado.
- `Gerente(x)`: x es un gerente.
- `Supervisa(x, y)`: x supervisa a y.

Función:

- `Superior(x)`: Devuelve el supervisor directo de x.

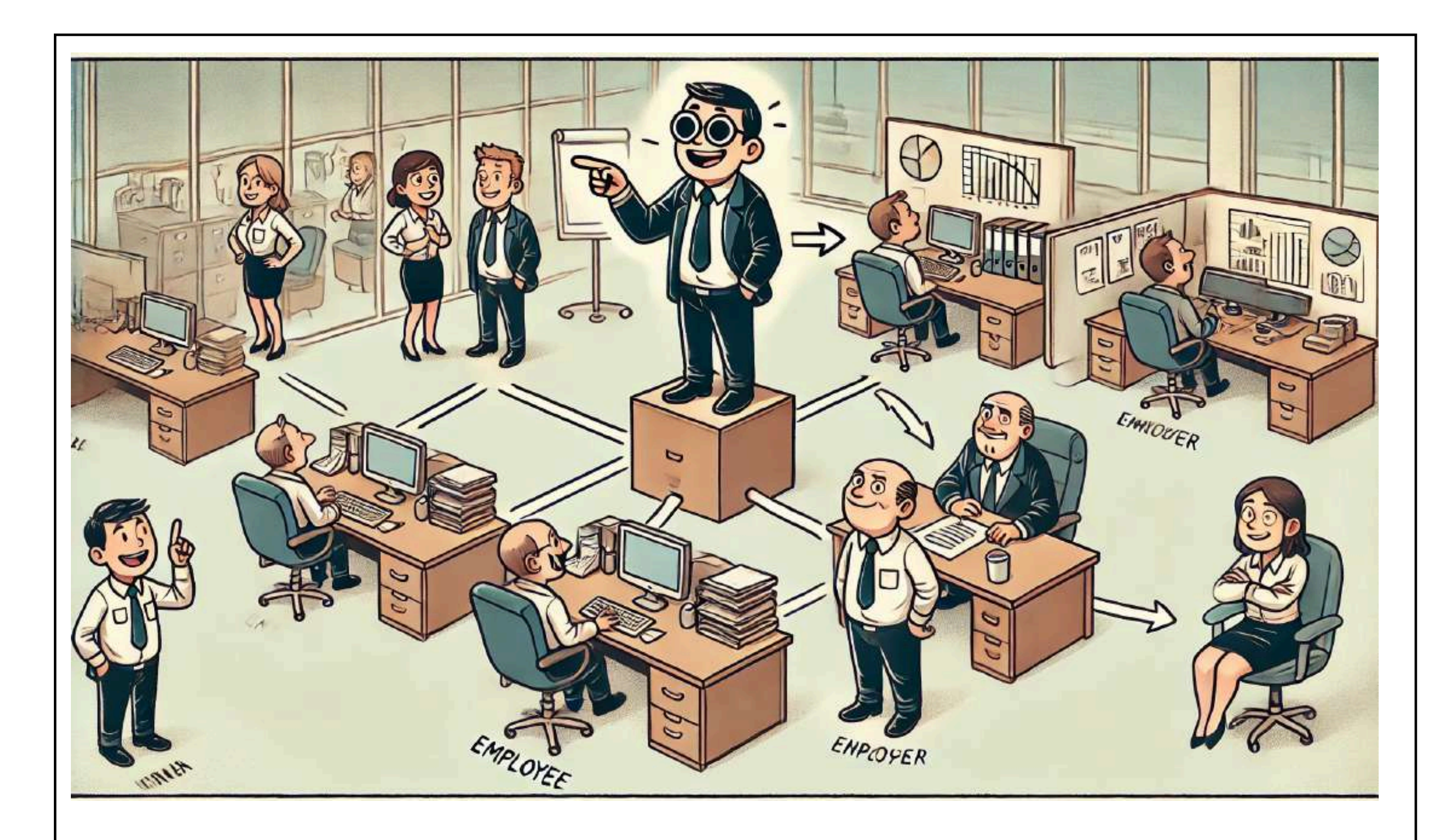
Reglas:

1. `∀x (Gerente(x) → Empleado(x))`: "Todos los gerentes son empleados."
2. `∃x ∃y (Gerente(x) ∧ Empleado(y) ∧ Supervisa(x, y))`: "Al menos un empleado es supervisado por un gerente."
3. `∀y (Empleado(y) → Gerente(Superior(y)))`: "El superior de un empleado es un gerente."
4. `∀x (Gerente(x) → ¬Supervisa(x, x))`: "Un gerente no puede supervisarse a sí mismo."

En un sistema de inteligencia artificial que gestiona roles y jerarquías dentro de una empresa, estas expresiones podrían utilizarse para:

- ✅ **Verificar la consistencia de los roles asignados**: asegurando que todas las personas marcadas como gerentes son también empleados.
- 🔍 **Inferir relaciones de supervisión**: si se conoce que alguien es un gerente, el sistema puede deducir automáticamente que supervisa a algún empleado.
- 🚫 **Asegurar restricciones**: como que ningún gerente se supervise a sí mismo, lo cual podría representar un conflicto de interés.

Este ejemplo muestra cómo la *lógica de primer orden* puede usarse para representar y razonar sobre roles y relaciones dentro de una estructura organizativa. Las aplicaciones en IA pueden variar desde sistemas de gestión de recursos humanos hasta la planificación de tareas en una empresa, donde es crucial entender y manejar la jerarquía y las relaciones de supervisión de manera lógica y consistente.



Aplicaciones en IA

Sistemas Expertos

Los sistemas expertos utilizan la lógica de primer orden para representar y manipular conocimientos especializados. Estos sistemas pueden realizar deducciones lógicas basadas en reglas y hechos expresados en LPO, lo que les permite emular el razonamiento de expertos humanos en dominios específicos.

Por ejemplo, en un sistema experto médico, podríamos tener reglas como:

```
∀x (TieneFiebre(x) ∧ TieneTos(x) → PosibleGripe(x))
```

Esta regla se leería como: "Para todo paciente x, si x tiene fiebre y tos, entonces x posiblemente tiene gripe". El sistema puede usar esta y otras reglas similares para hacer diagnósticos preliminares basados en los síntomas observados.

Planificación Automatizada

La planificación en IA utiliza la lógica de primer orden para representar estados del mundo, acciones y objetivos. Los planificadores automáticos utilizan estas representaciones para generar secuencias de acciones que logran objetivos específicos.

Por ejemplo, en un sistema de planificación de rutas para un robot, podríamos tener:

```
∀x ∀y (ConexionDirecta(x,y) ∧ Libre(y) → PuedeMoverseA(x,y))
```

Esta fórmula indica que si hay una conexión directa entre las ubicaciones x e y, y la ubicación y está libre, entonces el robot puede moverse de x a y. El planificador utilizaría estas reglas junto con el estado actual y el objetivo para generar un plan de movimiento.

Robótica

En robótica, la lógica de primer orden se utiliza para representar el conocimiento del robot sobre su entorno y para razonar sobre las acciones que puede tomar. Esto es especialmente útil en tareas de navegación, manipulación de objetos y toma de decisiones en entornos complejos.

Por ejemplo, un robot de almacén podría usar reglas como:

```
∀x ∀y (Objeto(x) ∧ UbicacionAlmacen(y) ∧ Vacio(y) → PuedeAlmacenarEn(x,y))
```

Esta regla indica que para cualquier objeto x y cualquier ubicación de almacén y, si y está vacía, entonces x puede ser almacenado en y. El robot utilizaría estas reglas junto con su percepción del entorno para decidir dónde colocar los objetos o cómo navegar por el almacén.

La lógica de primer orden tiene entonces un profundo impacto en los sistemas actuales, y especialmente en el campo de la inteligencia artificial. Por ejemplo, como ya comenté anteriormente, los sistemas de recomendación que utilizamos a diario en plataformas como Netflix o Amazon emplean principios de la lógica de primer orden para analizar nuestras preferencias y sugerir contenido relevante. Cuando estas plataformas infieren que *si a un usuario le gusta la ciencia ficción y las películas de acción, entonces probablemente disfrutará de 'Matrix'*, están aplicando reglas basadas en la lógica de primer orden.

Además, la lógica de primer orden es esencial en sistemas como los de consultas a bases de datos que son el corazón de muchas aplicaciones que usamos diariamente. Cuando hacemos una búsqueda en Google o filtramos productos en una tienda en línea estamos utilizando indirectamente la *lógica de primer orden*.

Por ejemplo, al buscar *restaurantes italianos abiertos ahora cerca mío*, el sistema está aplicando predicados lógicos para filtrar establecimientos que cumplan con todas estas condiciones al mismo tiempo.

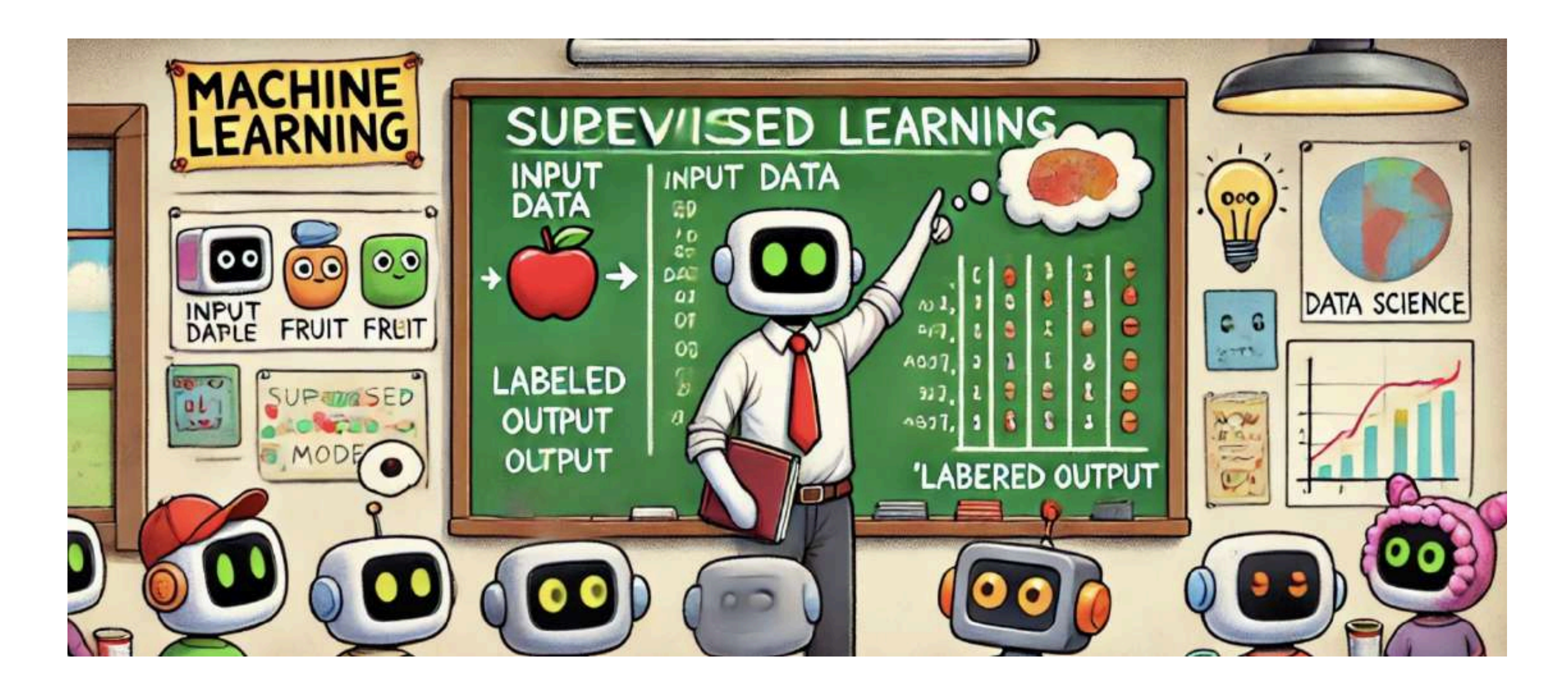
En el ámbito de la *domótica*, los sistemas inteligentes utilizan esta lógica para tomar decisiones basadas en múltiples variables, como ajustar la temperatura de una habitación considerando la hora del día, la presencia de personas y la temperatura exterior.



La lógica de primer orden, con su capacidad para expresar relaciones complejas y realizar inferencias, proporciona una base fundamental para lo que vamos a ver a continuación, que se trata de nada más ni nada menos que el famoso *machine learning*. Por ejemplo, cuando una red neuronal aprende a clasificar imágenes de animales, está esencialmente construyendo su propia versión de reglas lógicas (como "si tiene pelaje Y maúlla, ENTONCES es un gato"), **pero en lugar de utilizar reglas precisas y predefinidas, las aprende de los datos**.

De la misma manera, cuando un algoritmo de *machine learning* toma decisiones, está aplicando una forma más flexible y adaptativa de los principios lógicos que vimos recién, permitiendo manejar la incertidumbre y la ambigüedad que existe en el mundo real.

03. 🎯 Aprendizaje Supervisado (Machine Learning)



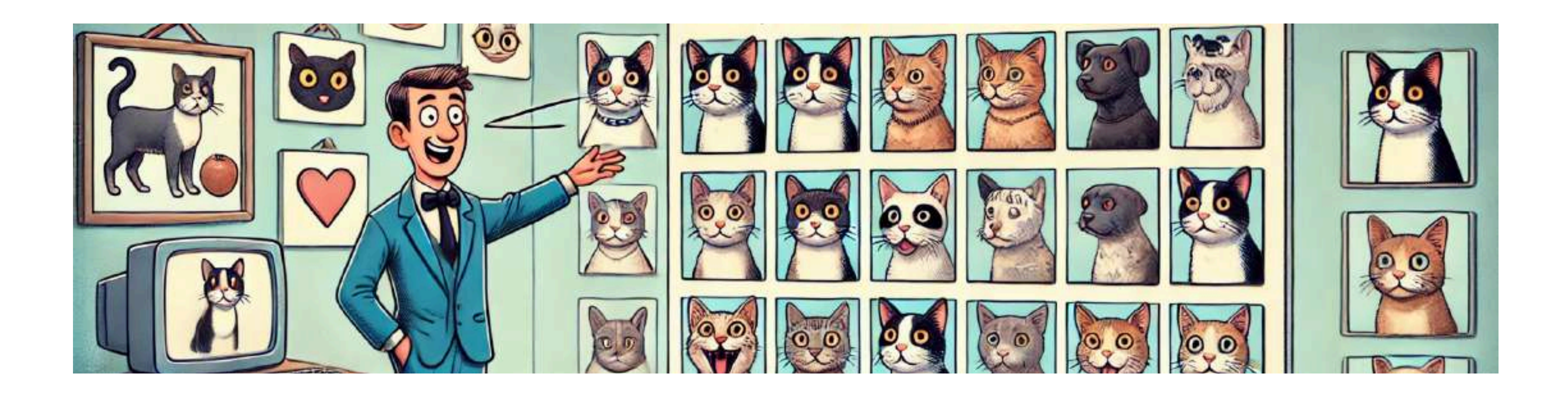
Llegamos ahora sí a uno de los tópicos de la Inteligencia Artificial más conocidos y famosos: el *aprendizaje automático,* mejor conocido como *Machine Learning.* La idea de poder enseñar a una *máquina* a "pensar" es sumamente atractiva y llama la atención y con justa razón. En el caso del *aprendizaje supervisado* lo que perseguimos es que, en lugar de programar un software[24] explícitamente para que haga una predicción o tome una decisión, lo que vamos a hacer es enseñarle ejemplos de eso que queremos que aprenda, y vamos a dejar que se *programe* y *aprenda* a hacerlo por sí mismo. Imaginate lo extremadamente difícil que sería programar desde cero un software que detecte imágenes de gatos. Este programa tendría que explícitamente codificar cómo detectar orejas, cómo detectar hocicos, como detectar bigotes, cómo detectar sus posturas. En definitiva, una imágen de un gato puede presentar una variabilidad casi infinita[25].

Por eso el aprendizaje supervisado es tan increíble, en lugar de hacer eso, simplemente le vamos a *enseñar* a un *algoritmo* muchísimas imágenes de gatos y el propio algoritmo decidirá qué patrones son los importantes a la hora de detectar gatos en imágenes y lo hará cada vez mejor mediante un proceso de optimización.

---

[24] El software es el conjunto de instrucciones y programas que le dicen a una computadora qué hacer, como por ejemplo Windows o las aplicaciones de tu celular. Es la parte que no puedes tocar físicamente (a diferencia de la pantalla o el teclado) pero que hace que todo funcione.

[25] Las "infinitas" variaciones que hacen difícil programar un detector de gatos incluyen: diferencias en el color del pelaje (pueden ser negros, blancos, atigrados, bicolores), diferentes posturas (acostados, sentados, estirados, en movimiento), distintos ángulos de la foto (frontal, perfil, desde arriba), diversas condiciones de iluminación, gatos parcialmente ocultos, diferentes razas con rasgos distintivos, expresiones faciales variadas, tamaños y edades diversos, y la presencia de elementos en el fondo que podrían confundir al programa. Cada una de estas variables tendría que ser programada explícitamente si no usáramos machine learning.



Parece magia pero no lo es.

El *aprendizaje supervisado* es entonces una variante del *machine learning* en la que los modelos[26] se entrenan utilizando datos *etiquetados*. Cuando hablamos de datos *etiquetados* nos referimos a un conjunto de datos[27] donde conocemos cuál es la *clase*[28] correcta o el valor numérico correspondiente. Por ejemplo, si queremos que nuestro modelo detecte actividad fraudulenta en transacciones bancarias, un *conjunto de datos (dataset)* etiquetado incluiría transacciones donde cada una de ellas podría estar marcada con un 1 o un 0, indicando *fraude* o *legítima* (las dos clases).

En otro caso de ejemplo, donde nuestro modelo debe entrenarse para predecir el precio de acciones, el *conjunto de datos* estaría compuesto por datos históricos de acciones, cada uno *etiquetado* con el precio de cierre de la acción en ese día específico.

En nuestro ejemplo de detección de gatos tendríamos un directorio en nuestra computadora lleno de imágenes de gatos y otro directorio con imágenes de cualquier otra cosa. Enseñaremos al modelo qué imágen tiene un gato y qué imágen no.

La razón de utilizar datos *etiquetados* es que permiten al modelo desarrollar la capacidad de aprender patrones y relaciones entre las características de esos datos y sus respectivas

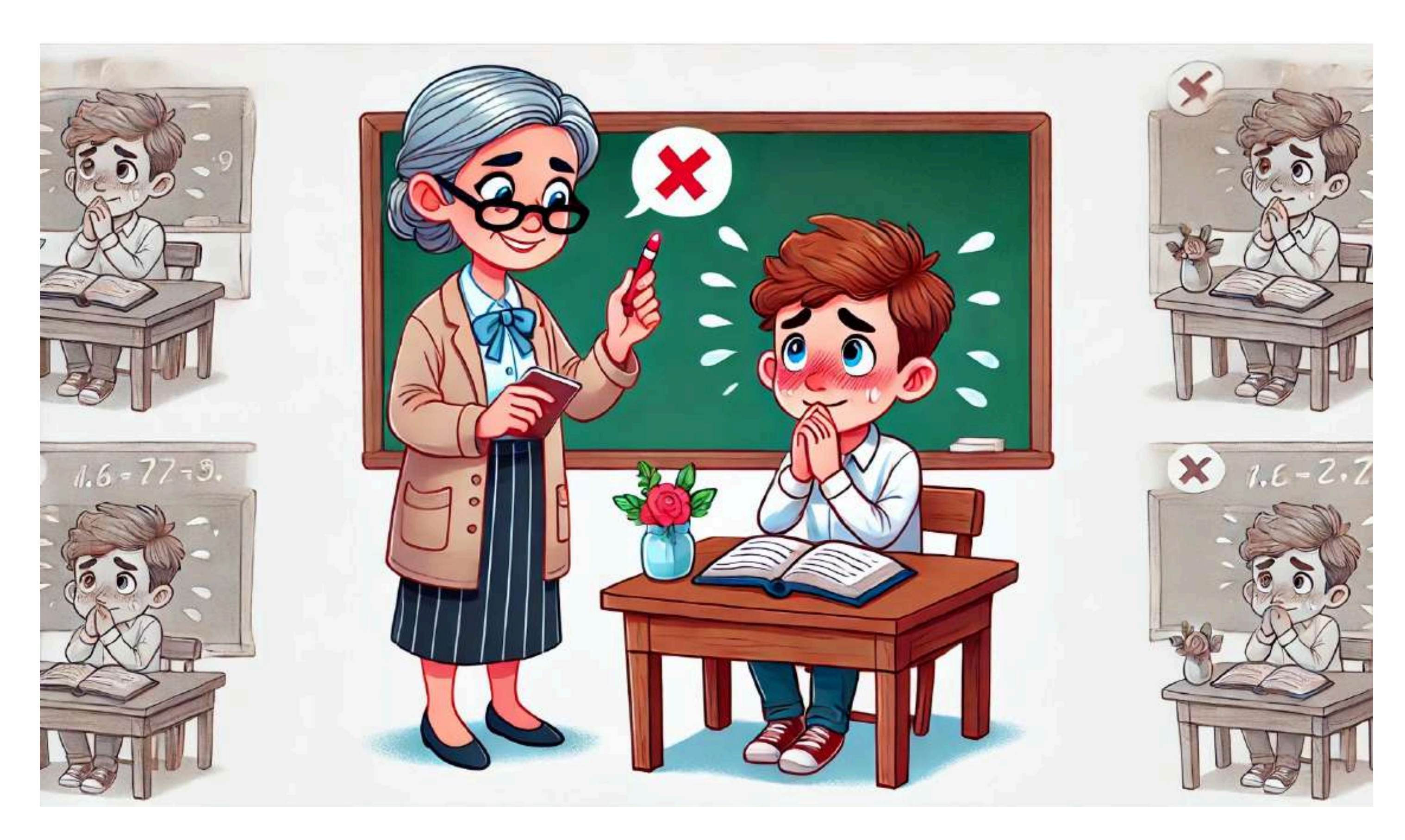
---

[26] En este libro utilizaré los términos "modelo" y "algoritmo" prácticamente como sinónimos para facilitar la lectura, aunque técnicamente tienen diferencias: un algoritmo es el procedimiento o conjunto de reglas que se siguen para resolver un problema (como el algoritmo de "árboles de decisión" o "regresión logística"), mientras que un modelo es la implementación específica de ese algoritmo entrenada con datos particulares. Por ejemplo, podemos usar el algoritmo de árboles de decisión para crear un modelo que detecte fraude bancario. En otras palabras, el algoritmo es la receta general, mientras que el modelo es el resultado final después de "cocinar" con nuestros datos específicos.

[27] En el contexto del machine learning supervisado, un "dato" se refiere a una unidad individual de información que el modelo utiliza para aprender. Puede ser una fila en una hoja de cálculo (donde cada columna es una característica), una imagen, una oración o un conjunto de mediciones en un punto del tiempo, dependiendo del tipo de problema que se esté abordando. El "etiquetado" es la información adicional que indica el resultado correcto o deseado para ese dato específico.

[28] En el contexto del machine learning supervisado, una clase es una categoría o etiqueta que asignamos a los datos. Por ejemplo, en el caso de detectar fraude bancario, tenemos dos clases: "fraude" y "legítima". En el caso de clasificar imágenes de animales, las clases podrían ser "gato", "perro", "pájaro", etc. Cada dato en nuestro conjunto de entrenamiento debe pertenecer a una (y solo una) de estas clases predefinidas.

etiquetas. ¿Cómo lo hace? Durante el entrenamiento, el modelo compara sus predicciones con las etiquetas correctas (también llamadas "ground truth") y ajusta iterativamente sus parámetros internos para reducir la diferencia entre sus predicciones y las etiquetas reales. De la misma manera que una persona aprende de sus errores cuando un profesor le señala que se equivocó en un examen, el modelo ajusta su "conocimiento" cuando sus predicciones no coinciden con las etiquetas correctas. Este proceso de retroalimentación continua es fundamental, ya que le permite al modelo "aprender" de sus errores y mejorar gradualmente su capacidad de generalización para poder hacer predicciones precisas sobre nuevos datos no vistos anteriormente.

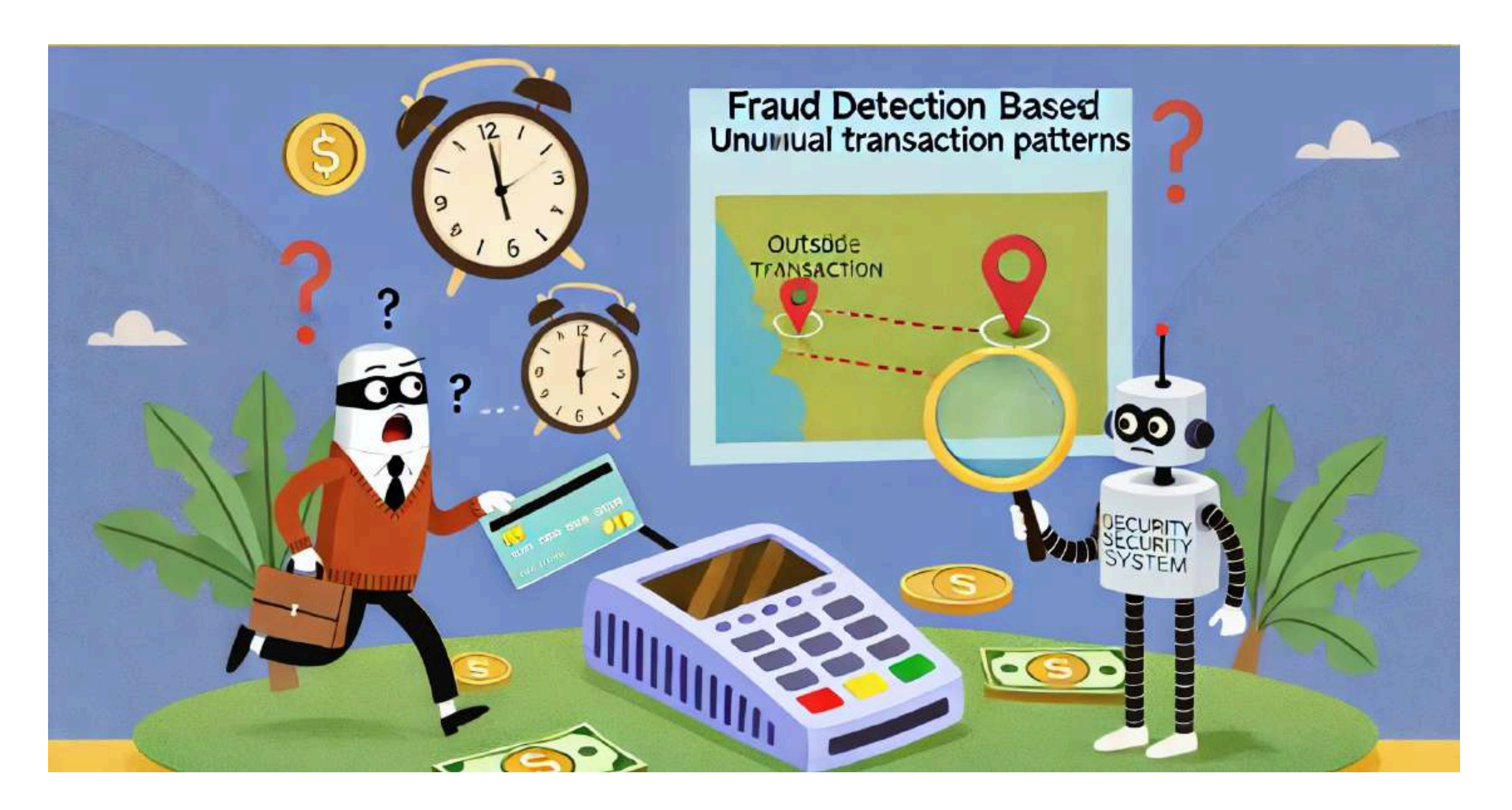


En este proceso de aprendizaje a partir de datos etiquetados, el componente fundamental que hace posible todo el proceso es el *algoritmo*[29]. Podemos pensar en el algoritmo como el "cerebro" del modelo, que define cómo este procesará la información de entrada, cómo ajustará sus parámetros internos durante el entrenamiento y cómo llegará a una decisión final. Por ejemplo, algunos algoritmos populares de aprendizaje supervisado incluyen la Regresion Logística, que es excelente para problemas de clasificación binaria, los Árboles de Decisión, que toman decisiones siguiendo una estructura similar a un diagrama de flujo, o las Redes Neuronales, que intentan imitar el funcionamiento de las neuronas en el cerebro

[29] Conjunto de pasos o instrucciones claras y bien definidas que se siguen para resolver un problema o realizar una tarea específica. Funciona como una receta que guía cómo alcanzar un resultado deseado, de manera eficiente y lógica, y puede aplicarse en una variedad de contextos, desde matemáticas hasta programación y la vida diaria.

humano. La elección del algoritmo adecuado depende de varios factores como el tipo de problema (clasificación o regresión), la cantidad y calidad de datos disponibles, y la complejidad de los patrones que necesitamos aprender.

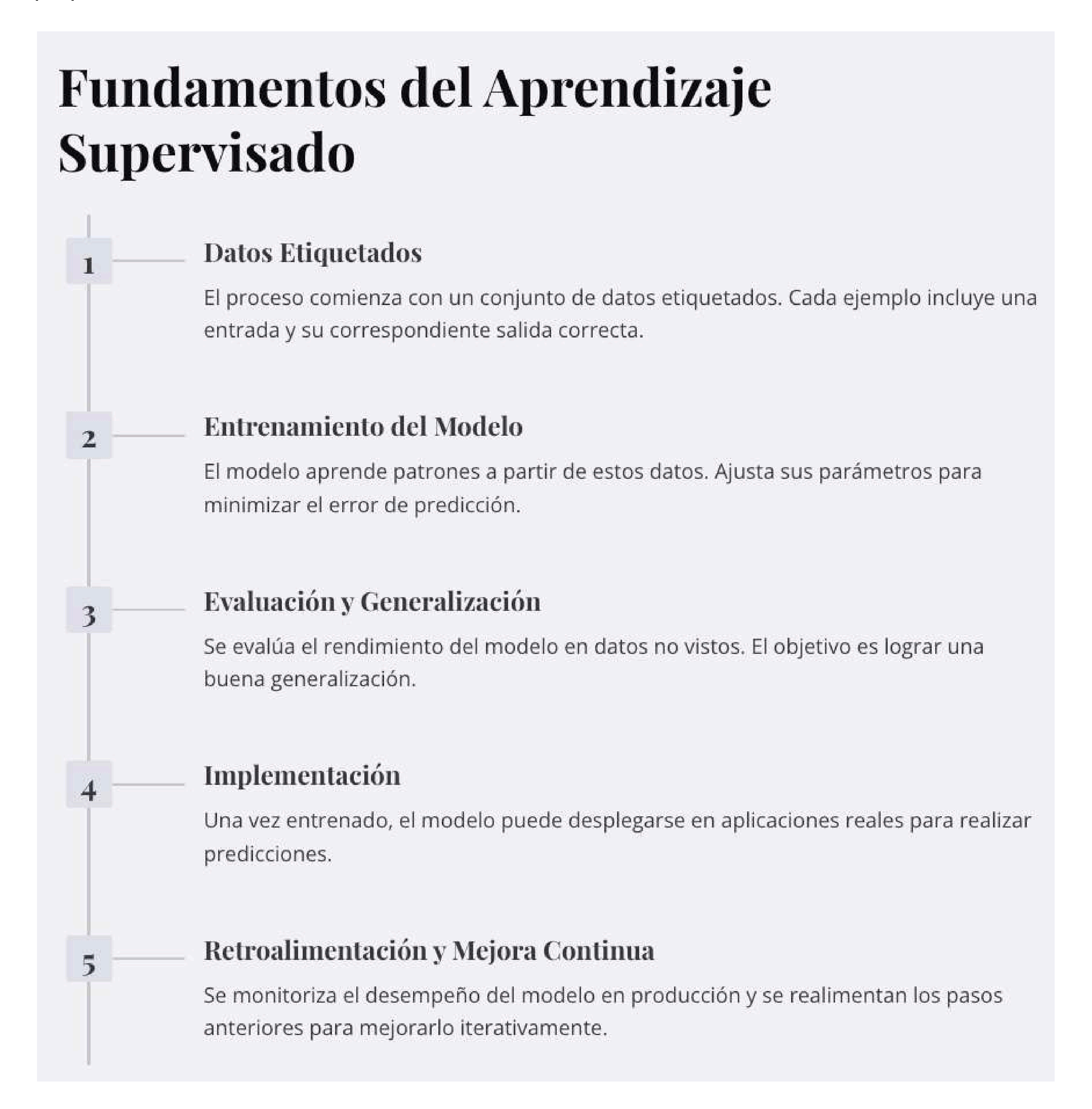
Veamos en detalle cómo funcionaría este proceso de aprendizaje en nuestro ejemplo de detección de fraude.



El *algoritmo* empieza analizando un gran volumen de transacciones bancarias *etiquetadas*, donde cada transacción está marcada como *fraudulenta* o *legítima* (o “1” y “0” como dijimos antes). Durante esta fase que llamamos de *entrenamiento*, el algoritmo realiza el trabajo de identificación de patrones y correlaciones que diferencian las transacciones fraudulentas de las legítimas, desarrollando gradualmente un modelo predictivo interno que le va a permitir realizar clasificaciones cada vez más precisas. Por ejemplo, el algoritmo podría *aprender* que cuando una transacción sucede fuera del horario habitual del cliente, tiene un monto significativamente superior al promedio, y se realiza desde una ubicación atípica, la probabilidad de que sea fraudulenta aumenta significativamente. Este tipo de patrones complejos y multidimensionales son exactamente lo que hace que el aprendizaje automático sea tan bueno: el algoritmo puede identificar y combinar distintos factores que tal vez no serían evidentes a simple vista para un analista humano o que serían demasiado complejos para programarlos manualmente.

Lo espectacular de este enfoque es que además es un proceso iterativo y adaptativo. A medida que el algoritmo procesa más datos, sigue ajustando sus parámetros internos, y por

lo tanto sigue refinando sus *criterios* de clasificación y mejorando su precisión. Esta capacidad de mejora continua es fundamental, especialmente en un contexto donde los patrones de fraude evolucionan constantemente y los estafadores desarrollan nuevas tácticas constantemente. El modelo puede adaptarse a estos cambios siempre que se le proporcionen nuevos datos de entrenamiento actualizados.



Pasemos ahora sí, a ver definiciones un poco más formales, así como nuevos conceptos clave y ejemplos concretos de *aprendizaje supervisado.*

El *aprendizaje supervisado*, como ya vimos, es una rama fundamental del *machine learning* que se centra en desarrollar modelos capaces de *aprender* patrones a partir de datos etiquetados. Este tipo de aprendizaje se puede aplicar a diversos tipos de información, desde el análisis de imágenes y texto hasta la predicción de valores numéricos como precios de viviendas o tendencias del mercado. La clave está en que el sistema aprende a partir de ejemplos donde ya conocemos la respuesta correcta.

El proceso se basa en utilizar un conjunto de *datos de entrenamiento*, donde cada ejemplo está compuesto por características de entrada y su correspondiente resultado esperado. El objetivo es que el algoritmo aprenda la relación entre estas entradas y salidas, desarrollando una función que pueda *mapear* correctamente nuevos datos. Esta función debe ser capaz de capturar los patrones subyacentes en los datos de manera efectiva.

Un aspecto crucial del aprendizaje supervisado es su capacidad de *generalización*. No alcanza con que el modelo funcione bien con los datos que usó para entrenarse; debe ser capaz de hacer predicciones precisas cuando se enfrente a datos nuevos y nunca antes vistos. Esta capacidad de generalización es lo que hace que los modelos de aprendizaje supervisado sean verdaderamente útiles en aplicaciones del mundo real, donde constantemente nos encontramos con nuevas situaciones y datos[30].

Conceptos clave del aprendizaje supervisado

- **Clasificación vs regresión**: Dentro del *aprendizaje supervisado*, existen dos grandes categorías de problemas: clasificación y regresión. En **clasificación**, el objetivo es asignar una etiqueta discreta a cada entrada (por ejemplo, determinar si una imagen contiene un gato o no, o si un correo electrónico es spam o no), mientras que en la regresión, buscamos predecir un valor numérico continuo (como el precio de una casa o la temperatura de mañana). Esta distinción es fundamental porque afecta tanto a la elección del algoritmo como a la forma en que evaluamos su rendimiento: mientras que en clasificación podemos hablar de *precisión* y *accuracy (exactitud)*[31], en regresión nos enfocamos en métricas como el *error cuadrático medio*[32].

---

[30] La capacidad de generalización es crucial en aplicaciones del mundo real. Por ejemplo: un sistema de detección de fraude bancario tiene que poder identificar nuevos tipos de estafas que los delincuentes inventen; un sistema de diagnóstico médico tiene que poder detectar enfermedades en pacientes con características diferentes a los del conjunto de entrenamiento; un sistema de reconocimiento facial tiene que poder identificar a una persona aunque cambie su peinado, use anteojos o envejezca; y un sistema de traducción automática tiene que poder traducir correctamente frases que nunca ha visto antes. Sin una buena generalización, estos sistemas serían inútiles en la práctica, ya que la realidad siempre presenta casos nuevos y diferentes.

[31] La precisión es una métrica específica que mide la proporción de predicciones positivas correctas sobre el total de predicciones positivas realizadas. Por ejemplo, si nuestro modelo clasifica 100 transacciones como fraudulentas, y 90 de ellas realmente eran fraudes, la precisión sería del 90%. Es importante distinguirla de la exactitud (accuracy), que mide la proporción total de predicciones correctas (tanto positivas como negativas) sobre el total de predicciones. En problemas de detección de fraude, la precisión es particularmente importante porque queremos minimizar los "falsos positivos" - transacciones legítimas incorrectamente clasificadas como fraudulentas - que podrían causar inconvenientes significativos a los clientes.

[32] El error cuadrático medio (ECM) es una métrica que mide qué tan lejos están las predicciones del modelo de los valores reales. Se calcula tomando las diferencias entre cada predicción y su valor real, elevándolas al cuadrado (para evitar que los errores positivos y negativos se cancelen entre sí), y luego promediándolas. Por ejemplo, si nuestro modelo predice que una casa vale $200,000 cuando en realidad vale $180,000, el error es de $20,000, y este valor se eleva al cuadrado antes de promediarlo con los demás errores. Cuanto menor sea el ECM, mejores son las predicciones del modelo.

Algunos ejemplos típicos de clasificación y de regresión son:

Clasificación

- **Diagnóstico médico:** ¿Es benigno o maligno?
- **Filtro de correo:** ¿Es spam o no spam?
- **Reconocimiento de imágenes:** ¿Gato, perro o pájaro?
- **Análisis de sentimientos:** ¿Positivo, negativo o neutral?
- **Evaluación crediticia:** ¿Aprobar o rechazar?
- **Control de calidad:** ¿Producto defectuoso o aceptable?

Regresión

- **Inmobiliaria:** Predicción del precio de una vivienda
- **Meteorología:** Pronóstico de temperatura
- **Finanzas:** Predicción del valor de acciones
- **Retail:** Estimación de ventas mensuales
- **Recursos Humanos:** Predicción de salario según experiencia
- **Energía:** Estimación del consumo eléctrico

- **Datos Etiquetados**: Los datos de entrenamiento consisten en pares de entrada y salida. Por ejemplo, en un problema de clasificación de imágenes, las entradas pueden ser imágenes y las salidas las etiquetas correspondientes que indican qué objeto está presente en cada imagen. En el caso de fraude, las entradas son los datos de la transacción y la salida es si esa transacción ha sido categorizada como *legítima* o *fraudulenta*.



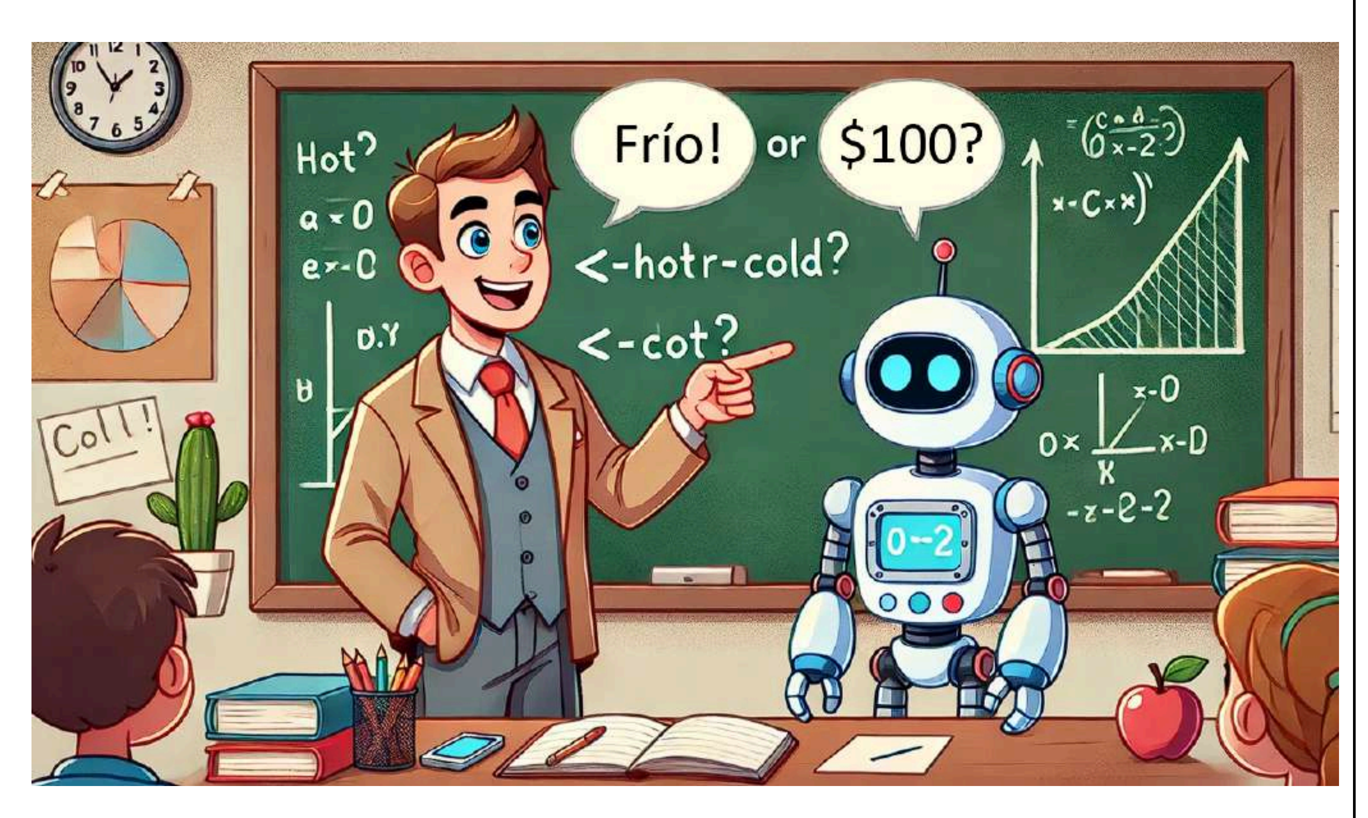
- **Función de pérdida**: Este concepto es mega-importante. Durante el entrenamiento el modelo hace predicciones con los datos de entrada y compara estas predicciones con las *etiquetas reales* utilizando una *función de pérdida* para cuantificar qué tanto se ha equivocado. La función de pérdida cuantifica el error entre las predicciones del modelo y las etiquetas reales.

  La función de pérdida es un componente fundamental en el aprendizaje supervisado. Imaginá que estás jugando a "adiviná el número". La *función de pérdida* sería como una puntuación que te diría qué tan lejos estás del número correcto.

  ¿Cómo funciona? El proceso es bastante directo:

  1. El modelo hace una predicción
  2. Comparamos esa predicción con el valor real
  3. La función de pérdida nos da un número que representa qué tan equivocada fue la predicción
  4. Cuanto más bajo sea este número, mejor es la predicción

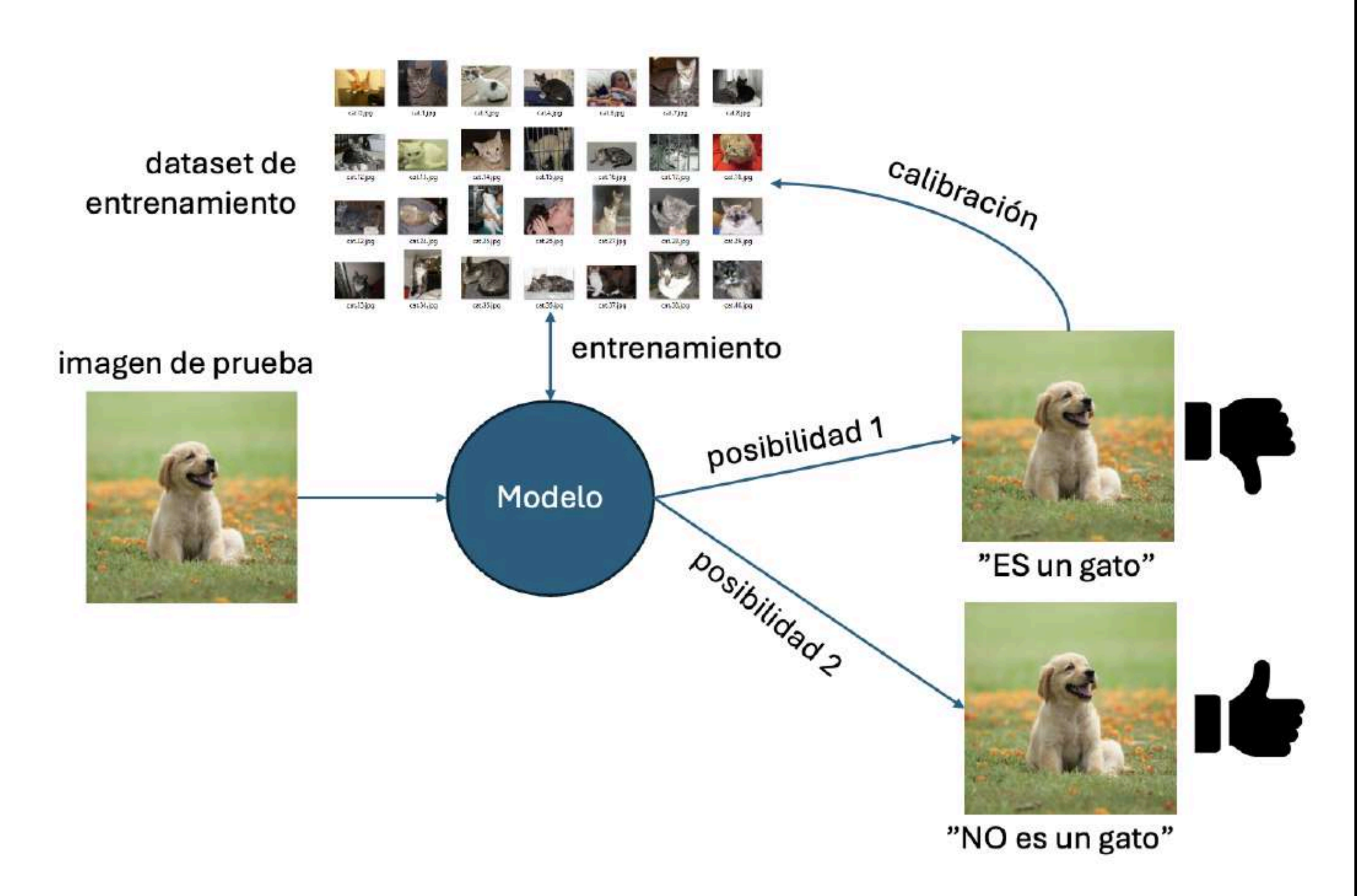
  ¿Por qué es importante? La función de pérdida guía al modelo para que mejore sus predicciones. Es como un maestro que le dice al modelo "caliente" o "frío" para ayudarlo a acercarse a la respuesta correcta.



  Este concepto es fundamental dentro del proceso de aprendizaje del algoritmo que discutimos anteriormente, ya que proporciona la retroalimentación necesaria para que el modelo pueda ajustar sus parámetros internos y mejorar su rendimiento durante el entrenamiento.

- **Optimización**: El proceso de entrenamiento implica **ajustar** los *parámetros* del modelo para minimizar el error que es medido por la función de pérdida (en simples términos: conseguir que el modelo se equivoque cada vez menos). Esto se logra mediante un **proceso de optimización** como el *descenso de gradiente* (que veremos más adelante). En la siguiente ilustración podemos ver cómo un modelo sería *entrenado* a partir de un conjunto de datos de entrenamiento donde le enseñamos una gran cantidad de las clases que debe aprender. Posteriormente el modelo se evalúa con imágenes

que no han sido parte del entrenamiento y se obtienen métricas de rendimiento. Los errores del modelo pueden servir para retroalimentar con nuevos datos y mejorar aún más.

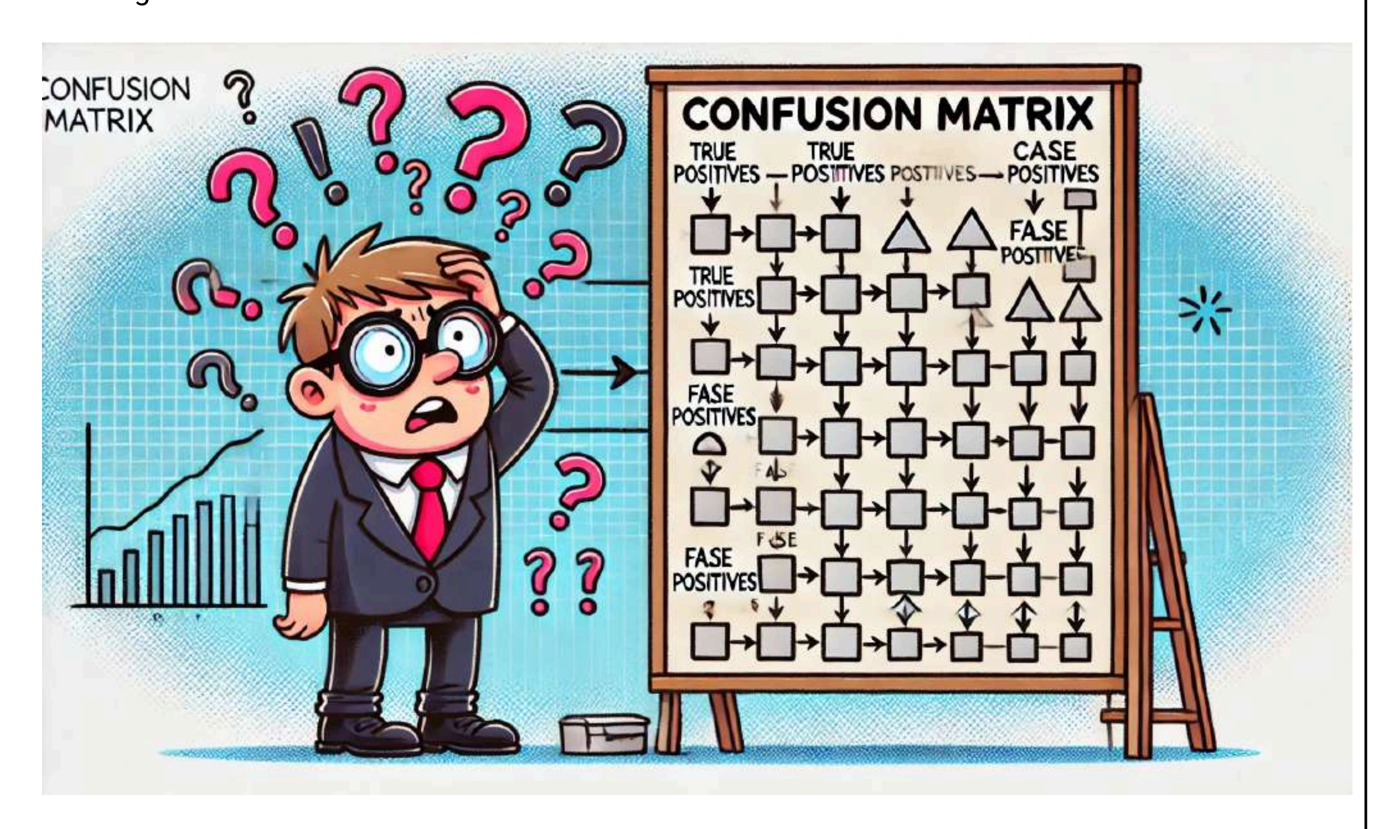


- **Matriz de Confusión:** Una matriz de confusión es una herramienta fundamental en la evaluación de modelos de *clasificación*[33], ya que nos permite visualizar y analizar el rendimiento del modelo en términos de *tipos de error*. En esta matriz, se muestran las predicciones correctas e incorrectas, lo que permite identificar cuatro posibles resultados: **verdaderos positivos** (cuando el modelo predice correctamente la clase principal, por ejemplo *fraude*), **verdaderos negativos** (cuando el modelo predice correctamente la ausencia de la clase principal, por ejemplo *no-fraude/legítima*), **falsos positivos** (cuando el modelo predice incorrectamente la presencia de la clase principal, por ejemplo *fraude* cuando en realidad era *legítima*) y **falsos negativos** (cuando el modelo no detecta la clase principal que está presente, por ejemplo que considere *legítima* una transacción que en realidad es *fraudulenta*). Estos tipos de error son cruciales para entender las debilidades de un modelo, ya que los falsos positivos y falsos negativos pueden tener diferentes consecuencias dependiendo del contexto, como en aplicaciones de detección de enfermedades, donde un falso negativo podría ser más crítico que un falso positivo. En nuestro ejemplo de detección de gatos, una matriz de confusión podría ofrecer este resultado a la hora de utilizar nuestro modelo entrenado con unas 100 imágenes de prueba:

[33] Recordá que por modelo de "clasificación" nos referimos a un modelo que debe aprender a distinguir entre clases. Por ejemplo gatos vs perros. O si la transacción es legítima o fraudulenta. Por el contrario, cuando leas sobre modelos de "regresión" se referirá a modelos que deben hacer una predicción numérica, por ejemplo un modelo para predecir el precio de las acciones de Microsoft de mañana, o el modelo de predicción de ventas de una empresa.

	Predicción del modelo: Gato	**Predicción del modelo**: NO Gato
Real: Gato	Verdaderos Positivos (TP): 50	Falsos Negativos (FN): 10
Real: No Gato	Falsos Positivos (FP): 5	Verdaderos Negativos (TN): 35

- **Verdaderos Positivos (TP)**: El modelo predijo correctamente 50 imágenes de gatos.
- **Falsos Negativos (FN)**: El modelo falló al identificar 10 gatos, prediciendo incorrectamente que no eran gatos.
- **Falsos Positivos (FP)**: El modelo identificó incorrectamente 5 imágenes que no eran gatos como si lo fueran.
- **Verdaderos Negativos (TN)**: El modelo predijo correctamente 35 imágenes que no eran gatos.



Este ejemplo de **matriz de confusión** muestra el desempeño del modelo al clasificar correcta e incorrectamente las imágenes. A partir de esta matriz se pueden calcular métricas como **precisión**, **sensibilidad**, **especificidad**, entre otras[34]. Esto es únicamente aplicable a los modelos de aprendizaje supervisado que llamamos de "clasificación", como en este caso donde nuestro objetivo era clasificar imágenes. En el caso de predicciones numéricas (por ejemplo, predecir el precio de acciones) esto sería un modelo de "regresión" y el error en *regresión* no puede medirse con una matriz de confusión sino utilizando métricas como el Error Cuadrático Medio (MSE), que cuantifica la diferencia entre los valores predichos y los valores reales, proporcionando una medida del

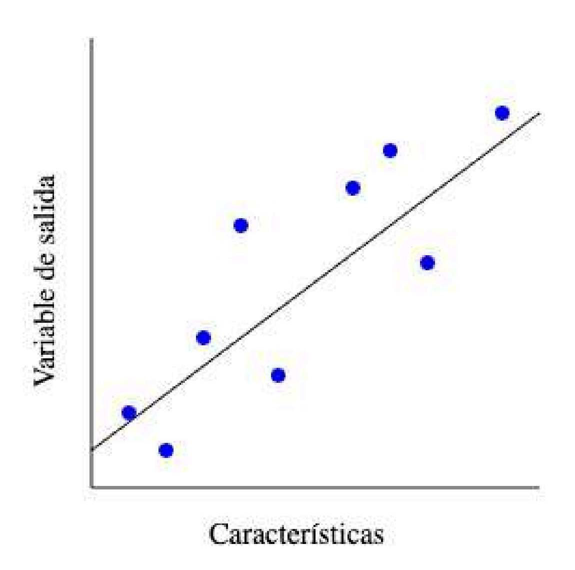
[34] *Existen métricas más avanzadas como las curvas ROC (Receiver Operating Characteristic) y el área bajo la curva ROC (AUC) que permiten evaluar el rendimiento del modelo a diferentes umbrales de clasificación. La curva ROC visualiza la relación entre la tasa de verdaderos positivos y falsos positivos a medida que se ajusta el umbral de decisión del modelo, mientras que el AUC proporciona un valor único entre 0 y 1 que resume el rendimiento general del modelo (donde 1 representa un clasificador perfecto). Sin embargo, estos conceptos exceden el alcance introductorio de este libro.*

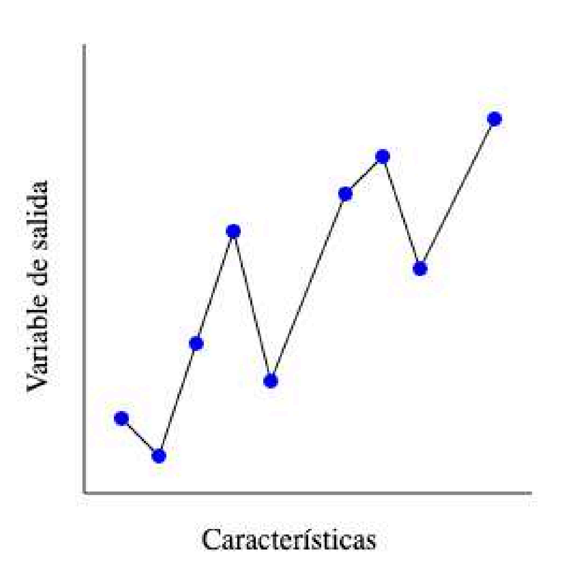
desempeño del modelo. Recordá entonces que la matriz de confusión es aplicable en *clasificación* y no en *regresión.*

- **Generalización**: Como se comentó anteriormente, el objetivo final del aprendizaje supervisado es que nuestros modelos no solo funcionen bien con los datos de entrenamiento, sino que también *generalicen* bien a nuevos datos no vistos durante el proceso de entrenamiento. Con las siguientes imágenes podemos ver el clásico ejemplo de lo que se denomina *overfitting*[35]. Cada punto azul en los gráficos representa un dato de entrenamiento (es decir, que el modelo lo *ha visto*) y las líneas de cada gráfico podrían ser dos modelos diferentes que hemos conseguido durante nuestra fase de experimentación. Un modelo como el de la derecha, que pase por cada punto azul a priori parecería un excelente modelo (después de todo, el "Error de Entrenamiento" es 0), pero lo único que estaría haciendo es *memorizar* los datos de entrenamiento. Enseguida vamos a ver cómo este modelo es en realidad terrible. Nuestro objetivo nunca puede ser un modelo que *memorice* datos sino uno que sirva para nuevos datos (otros puntos que no sean los azules) que vendrán en el futuro cuando lo usemos.



Overfitting (Entrenamiento)





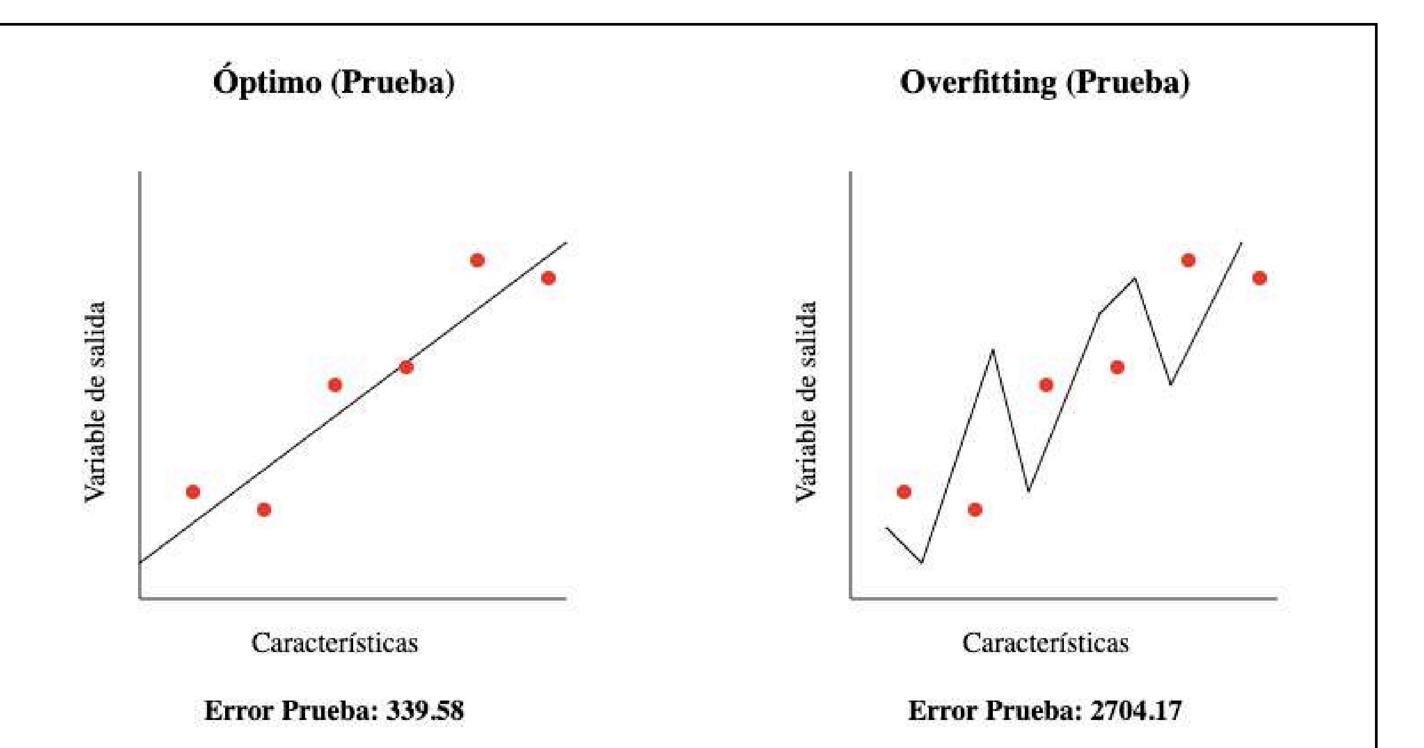
Error Entrenamiento: 1138.89

Error Entrenamiento: 0.00

En ese sentido, un modelo lineal como el de la izquierda tiene muchas más chances de funcionar mejor con datos nuevos que el modelo de la derecha a pesar de que su "Error de Entrenamiento" es positivo.

Podemos ver ahora qué pasaría con nuestros dos candidatos cuando los probamos con datos nuevos. Imaginá que luego de entrenar nuestros dos modelos basados en los datos azules decidimos recopilar nuevos datos y los representamos en rojo. En la siguiente nueva ilustración, los puntos rojos son nuevos, pero las líneas (modelos) son exactamente los mismos que en el gráfico con los puntos azules.

[35] *Sobreajuste*, es decir, se ajusta perfectamente a los datos de entrenamiento pero tiene nula utilidad con datos nuevos.



Efectivamente, ahora sí podemos ver como el modelo lineal de la izquierda termina teniendo un error mucho más bajo que nuestro modelo *sobreajustado* a los datos azules de entrenamiento. Entonces es evidente que el modelo que parecía ser *perfecto* con los datos azules de entrenamiento, con un error de 0, evaluado con datos nuevos (los rojos) nos deja ver cómo en realidad era *perfectamente malo*, mientras que el modelo que parecía más simple por ser una línea recta y tenía un cierto error incluso con los datos de entrenamiento, vemos que con datos nuevos no lo hace nada mal, y eso lo convierte en un *buen modelo*[36].

Existen muchas técnicas que un científico de datos aplica para evitar el *overfitting* pero eso excede al alcance de este libro[37].

Finalmente, esto nos viene muy bien para pasar ahora a hablar de algoritmos que típicamente se utilizan en *machine learning supervisado* y vamos a empezar con nuestro modelo que acabamos de ver, el modelo lineal simple, mejor conocido como *regresión lineal*.

[36] Recordá entonces que un "buen modelo" no necesariamente es aquel que tiene el menor error posible, sino el que mantiene un error razonable y consistente tanto con datos conocidos como con datos nuevos. De hecho, un modelo con error cero en los datos de entrenamiento suele ser una señal de alarma que sugiere memorización en lugar de aprendizaje.

[37] Algunas técnicas comunes para prevenir el *overfitting* incluyen: *regularización* (que penaliza la complejidad excesiva del modelo), *dropout* (que desactiva aleatoriamente algunas conexiones durante el entrenamiento), *early stopping* (que detiene el entrenamiento cuando el rendimiento en datos de validación deja de mejorar), y *validación cruzada* (que evalúa el modelo en diferentes subconjuntos de datos). También es común usar conjuntos de datos más grandes y realizar *data augmentation* (aumentar artificialmente los datos de entrenamiento mediante transformaciones).

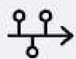
Algoritmos populares del aprendizaje supervisado

Algoritmos Populares en Aprendizaje Supervisado



Regresión Lineal

Predice valores continuos. Útil para pronósticos y análisis de tendencias.



Máquinas de Vector Soporte

Algoritmo robusto para clasificación y regresión. Eficaz en espacios de alta dimensión.



Árboles de Decisión

Modelos versátiles para clasificación y regresión. Fáciles de interpretar y visualizar.



Redes Neuronales

Capaces de capturar patrones complejos. Ideales para problemas de alta dimensionalidad.

- **Regresión Lineal:** es una técnica estadística utilizada para predecir un valor continuo[38] basándose en una o más variables independientes[39]. Por ejemplo, se podría utilizar para predecir el precio de una casa en función de sus características. Una ecuación típica de regresión lineal para este caso podría ser:

  Precio = 100,000 + 1,000 × metros_cuadrados + 50,000 × número_baños - 5,000 × años_antigüedad + ε,

  Donde ε representa el error del modelo.

  Para entender mejor cómo funciona este modelo, consideremos estos datos de ejemplo:

Casa	Metros cuadrados	Número de baños	Años de antigüedad	Precio real
1	100	2	10	235,000
2	85	1	15	180,000
3	120	2	5	275,000
4	95	1	8	220,000
5	150	3	2	350,000

  A partir de datos como estos, el algoritmo de regresión lineal 'aprende' o se 'entrena' encontrando los

[38] Un valor continuo es aquel que puede tomar cualquier valor dentro de un rango, como la temperatura (23.1°C, 23.15°C, 23.153°C...), el precio ($100.50, $100.51...) o la altura (1.75m, 1.756m...). Esto contrasta con los valores discretos o categóricos que solo pueden tomar valores específicos y limitados, como el día de la semana (lunes, martes...) o el tipo de vivienda (casa, departamento, dúplex).

[39] Decimos que una variable es independiente cuando su valor no está determinado por otras variables del sistema o experimento, sino que puede ser elegido o manipulado libremente por el investigador para estudiar su efecto sobre otras variables.

mejores coeficientes (los números que multiplican a cada variable en nuestra ecuación) que minimicen el error entre las predicciones y los precios reales. Este proceso se realiza típicamente utilizando un método llamado *descenso del gradiente*, que lo veremos más adelante, y que ajusta iterativamente estos coeficientes hasta encontrar la combinación que produce las mejores predicciones posibles para los datos de entrenamiento. En otras palabras, el algoritmo prueba diferentes combinaciones de coeficientes hasta encontrar aquellos que hacen que la ecuación prediga valores lo más cercanos posible a los precios reales de las casas en nuestros datos de entrenamiento.

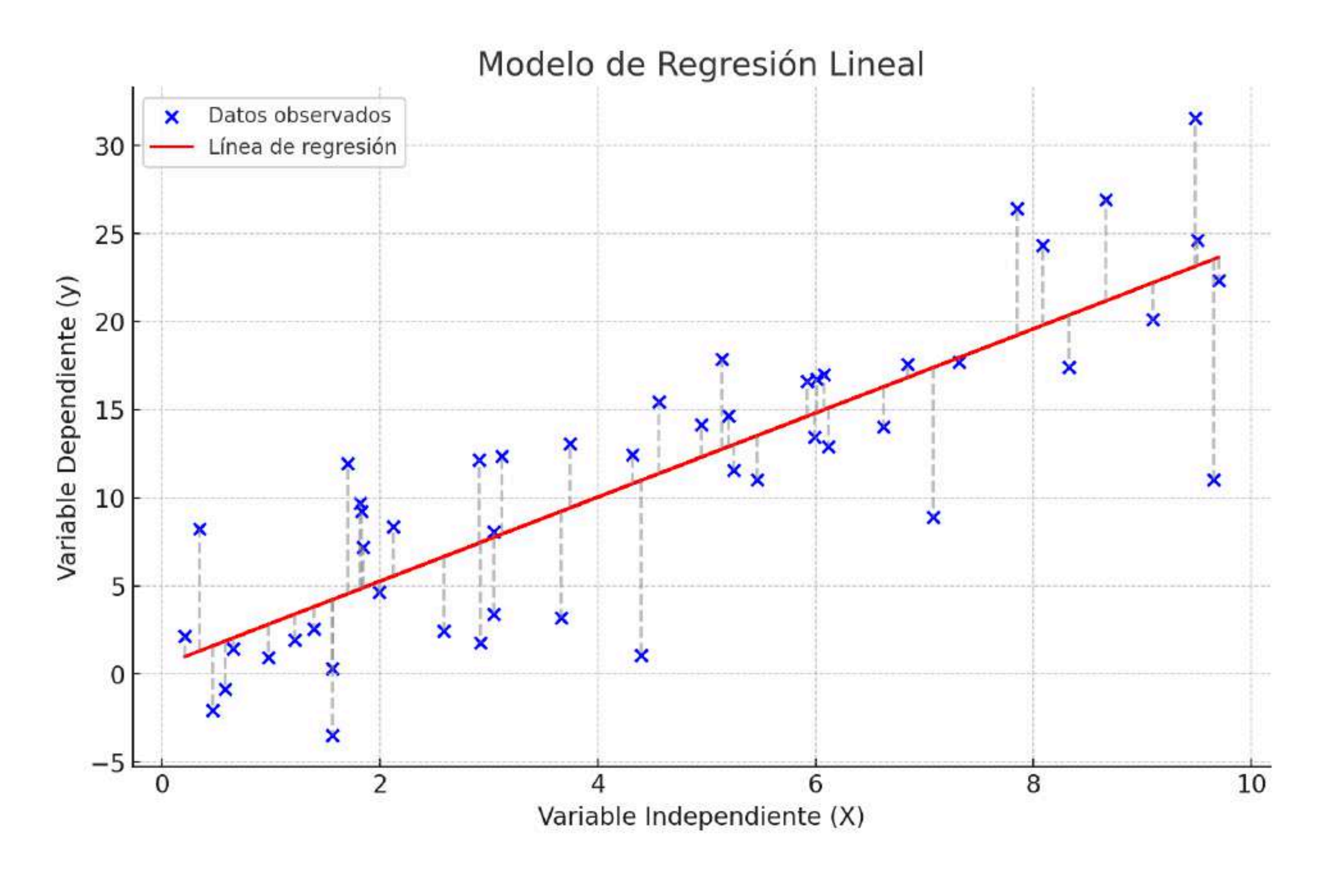
Así, por ejemplo, una casa de 100 metros cuadrados, con 2 baños y 10 años de antigüedad tendría un precio estimado de:

100,000 + 1,000 × 100 + 50,000 × 2 - 5,000 × 10 = 240,000 unidades monetarias

En este ejemplo, bastante cercano al precio real de 235,000.

Como podés ver este método establece una relación lineal entre la variable a predecir (variable dependiente) y las variables predictoras (variables independientes). En el caso de la predicción del precio de una casa, las variables independientes podrían incluir el tamaño de la propiedad, el número de habitaciones, la ubicación, y la antigüedad, entre otras.

La regresión lineal busca entonces encontrar la mejor línea recta (en el caso simple) o un hiperplano (en casos multidimensionales) que minimice la diferencia entre los valores predichos y los valores reales observados.



Para encontrar esta "mejor línea", el método analiza los datos observados y busca un balance que minimice los errores de predicción. Por ejemplo, si queremos predecir el precio de una casa (variable dependiente) basándonos en su tamaño (variable independiente), la regresión lineal va a trazar una

línea que mejor se ajuste a los datos disponibles de precios y tamaños de casas similares. Esta línea nos va a permitir estimar precios para casas de diferentes tamaños, aunque tenemos que recordar que es una aproximación y pueden existir otros factores influyentes no considerados en el modelo.

En el gráfico anterior podemos ver claramente cómo funciona:

- Los **puntos azules (×)** representan los datos reales observados, es decir, son pares de valores para los cuales conocemos tanto la variable independiente (x) como la dependiente (y). En un contexto real, cada punto podría representar, por ejemplo, una venta previa de una casa donde conocemos tanto su tamaño (x) como el precio final de venta (y). Estos datos históricos son fundamentales porque son la "verdad conocida" a partir de la cual el modelo aprenderá los patrones subyacentes en los datos. Mientras más puntos tengamos (es decir, mientras más datos históricos tengamos), mejor podrá ser el ajuste de nuestro modelo, siempre y cuando estos datos sean representativos del fenómeno que queremos modelar.
- La **línea roja** representa la "mejor línea" que la regresión lineal calculó mediante el proceso de entrenamiento. Esta línea es el resultado de encontrar los coeficientes óptimos (pendiente e intercepto) que minimizan el error total entre las predicciones y los valores reales. En términos matemáticos, el algoritmo busca minimizar la suma de los cuadrados de las distancias verticales entre cada punto azul y la línea roja (esto se conoce como "mínimos cuadrados"). La *pendiente* de esta línea nos indica cuánto cambia la variable dependiente (y) por cada unidad que aumenta la variable independiente (x), mientras que el punto donde la línea cruza el eje *y* (la intercepción) nos da el valor base cuando *x* es cero. Es importante notar que "mejor línea" no significa que sea perfecta, sino que es la que mejor balancea los errores positivos y negativos para todos los puntos en nuestro conjunto de datos.
- Aunque no todos los puntos caen exactamente sobre la línea roja, esta línea captura la tendencia general: a medida que la variable independiente (x) aumenta, la variable dependiente (y) también tiende a aumentar.
- Los puntos que están alejados de la línea roja representan las desviaciones o errores del modelo, pero la línea se posiciona de manera que minimice estas desviaciones en general.
- La pendiente positiva de la línea roja nos indica que existe una relación positiva entre las variables: cuando una aumenta, la otra también tiende a aumentar.

La regresión lineal, a pesar de su aparente simplicidad, es una herramienta muy potente que nos introduce a conceptos fundamentales del *machine learning*: la idea de entrenar un modelo con datos históricos, la importancia de minimizar errores de predicción, y la capacidad de hacer predicciones sobre nuevos casos. Si bien existen modelos más sofisticados que pueden capturar relaciones más complejas entre variables, la regresión lineal sigue siendo una técnica esencial en el conjunto de herramientas de cualquier científico de datos, no solo por su utilidad práctica en casos donde las relaciones son aproximadamente lineales, sino también porque sus principios básicos son la base para entender algoritmos más avanzados. Además, computacionalmente es muy eficiente tanto en su entrenamiento como en su uso para hacer predicciones, lo que la hace especialmente útil cuando trabajamos con grandes volúmenes de datos o necesitamos respuestas en tiempo real.

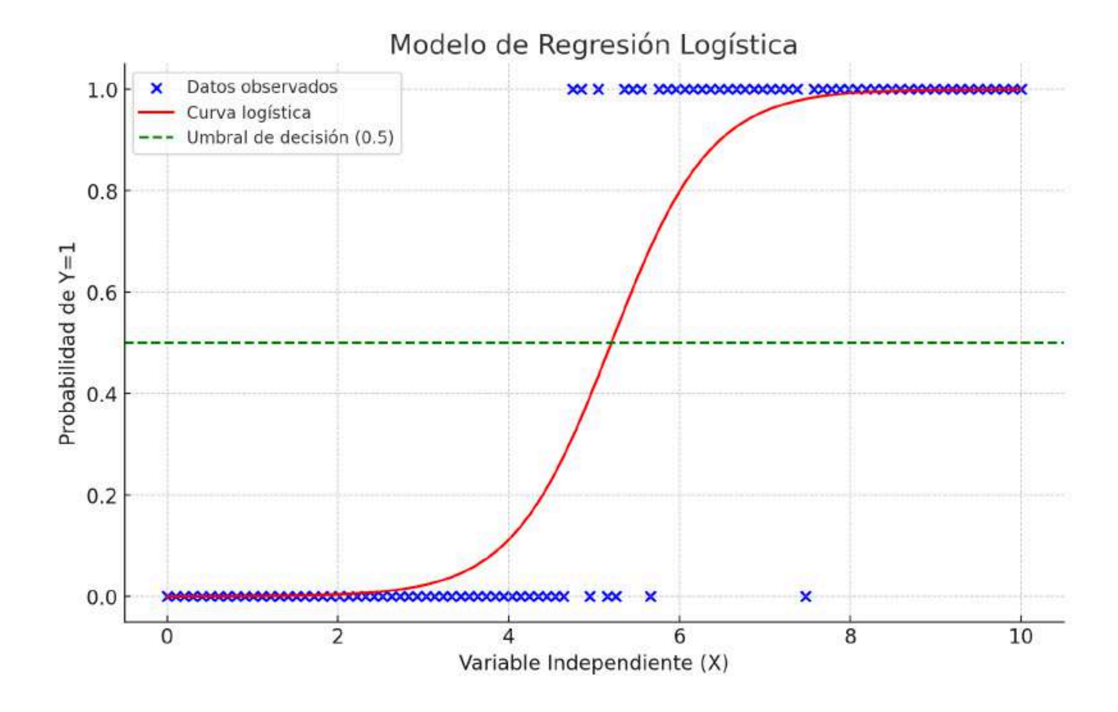
- **Regresión Logística**: La regresión logística, a pesar de su nombre que sugiere similitud con la regresión lineal, es en realidad uno de los pilares fundamentales del aprendizaje automático para problemas de clasificación, donde el objetivo suele ser predecir si algo pertenece a una de múltiples categorías posibles.

  Esta técnica transforma un problema de clasificación en una estimación de probabilidades. Por ejemplo:

  - **En medicina**: 🏥 predecir si un paciente tiene una enfermedad (✅/❌) basándose en sus síntomas y resultados de pruebas

- **En banca**: 🏦 decidir si aprobar un préstamo (✔️/❌) analizando el historial crediticio del cliente
- **En marketing**: 🛍️ predecir si un cliente comprará un producto (🛒/❌) según su comportamiento previo
- **En seguridad informática**: 🔒 detectar si una transacción es fraudulenta (⚠️/✅) basándose en patrones de comportamiento

A diferencia de la regresión lineal que predice valores continuos (como precios o temperaturas), la regresión logística transforma sus predicciones a valores entre 0.0 y 1.0, que representan la probabilidad de pertenecer a una categoría. Por ejemplo, un resultado de 0.8 indicaría un 80% de probabilidad de pertenecer a la categoría positiva (como ser *spam*, tener la *enfermedad* o *aprobar* el crédito).



La regresión logística se puede entender claramente a través del gráfico superior, que muestra cómo funciona la clasificación binaria.

El gráfico nos muestra varios elementos clave:

- Los **puntos azules (×)** representan los datos observados que solo pueden tomar dos valores: 0 o 1 (por ejemplo, no-spam o spam)
- La **línea roja** es la "curva logística" que modela la probabilidad de pertenecer a la clase 1
- La **línea verde punteada** marca el umbral de decisión (0.5):
  - Si la curva predice un valor mayor a 0.5, clasificamos el caso como clase 1
  - Si predice menor a 0.5, lo clasificamos como clase 0
- La forma de "S" de la curva es característica de la regresión logística:
  - En los extremos, la probabilidad se acerca a 0 o 1 pero nunca los sobrepasa
  - En el centro (alrededor de x=6 en este caso) es donde ocurre la transición más pronunciada entre las clases

Por ejemplo, si esto fuera un modelo para detectar ✉️ *spam*, los correos con características que los ubiquen a la derecha del gráfico (x>6) tendrían una alta probabilidad de ser clasificados como 🚫

spam, mientras que los de la izquierda (x<6) serían clasificados como ✅ correos *legítimos*.

En la práctica, la regresión logística la utilizamos frecuentemente como primer modelo de referencia (o "baseline") en problemas de clasificación, debido a su rapidez de entrenamiento y facilidad de interpretación. Los científicos de datos solemos empezar con este modelo para establecer un punto de comparación (*benchmarking*) antes de probar algoritmos más complejos. Su capacidad para proporcionar no sólo una clasificación sino también una probabilidad es especialmente útil en aplicaciones del mundo real donde las consecuencias de las decisiones son importantes: por ejemplo, en diagnósticos médicos donde necesitamos diferentes umbrales de decisión según si queremos minimizar los falsos positivos o los falsos negativos, o en detección de fraude donde podemos ajustar el umbral según el balance deseado entre seguridad y conveniencia del usuario. Esta flexibilidad, combinada con su eficiencia computacional, hace que la regresión logística siga siendo una herramienta fundamental en el arsenal de cualquier científico de datos moderno.

- **Árboles de Decisión**: son modelos de *machine learning* que funcionan tomando decisiones paso a paso, similar a cómo los humanos seguimos un diagrama de flujo.

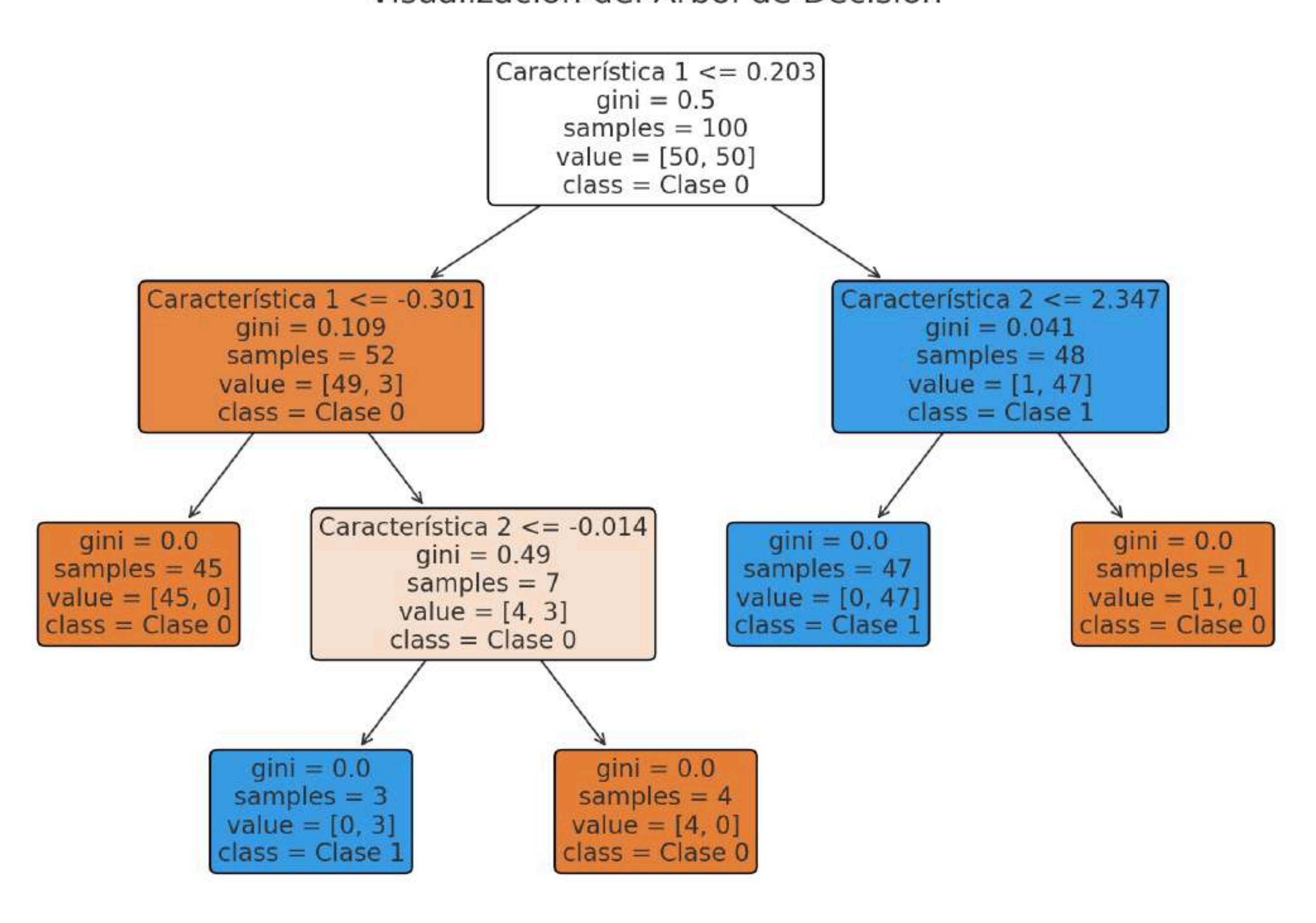


Imaginemos un árbol de decisión como un juego de preguntas donde cada respuesta nos lleva a una nueva pregunta hasta llegar a una conclusión final. Por ejemplo, para clasificar si un cliente representa un riesgo crediticio:

1. Primera pregunta: "¿Es el ingreso mensual mayor a $3000?"
   - Si la respuesta es SÍ → vamos a la siguiente pregunta
   - Si la respuesta es NO → hacemos una pregunta diferente
2. Segunda pregunta: "¿Tiene más de 2 años en su trabajo actual?"
   - Si la respuesta es SÍ → bajo riesgo crediticio
   - Si la respuesta es NO → "¿Tiene garantías?"

Y así sucesivamente hasta llegar a una decisión final. La ventaja principal de este método es que es fácil de entender y explicar, ya que podemos seguir el camino de decisiones que llevó a cada

predicción. Sin embargo, si el árbol se hace demasiado complejo (con muchas ramas y condiciones), puede empezar a *memorizar*[40] los datos en lugar de aprender patrones generales, lo que llamamos "sobreajuste". Acá podes ver un ejemplo visual de un árbol de decisión:



Este árbol de decisión se interpreta siguiendo un camino desde la raíz hasta una hoja. Por ejemplo, comenzamos en el nodo superior que evalúa si la Característica 1 es menor o igual a 0.203. Si es así, nos movemos al nodo izquierdo que verifica si la misma Característica 1 es menor o igual a -0.301. Suponiendo que no lo es, pasamos al nodo que examina si la Característica 2 es menor o igual a -0.014. Si esta última condición no se cumple, llegamos a un nodo hoja final que clasifica la muestra como Clase 0, basándose en 4 muestras con características similares. Este proceso ilustra cómo el árbol utiliza múltiples decisiones basadas en los valores de las características para determinar la clasificación final de una muestra dada.

Supongamos que tenemos un árbol de decisión para predecir el precio de las casas según sus características. El árbol se ve así:

[40] Por ejemplo, el árbol podría crear una rama específica que diga "si el ingreso es exactamente $3,127.54 Y tiene exactamente 2.3 años en el trabajo Y vive en la calle Principal número 123, entonces es bajo riesgo". Esta regla tan específica probablemente solo aplicaría a un único caso en los datos de entrenamiento y no sería útil para evaluar nuevos clientes, ilustrando cómo el modelo "memoriza" en lugar de "aprender" patrones generales útiles.

Vamos a interpretar un ejemplo específico utilizando este árbol de decisión.

Supongamos que tenemos una casa con las siguientes características:

- Metros cuadrados: 100 m²
- Habitaciones: 3
- Garaje: No
- Estación de Subte/Metro cerca: Sí
- Ubicación: Afueras (fuera de la ciudad)



Paso a paso por el árbol:

1. **Nodo raíz:**
   i. El árbol comienza dividiendo por metros cuadrados.
   ii. Como nuestra casa tiene 100 m², que es menor o igual a 137.5, seguimos la rama izquierda.
2. **Siguiente nodo:**
   i. En este nivel, el árbol evalúa habitaciones.
   ii. Nuestra casa tiene 3 habitaciones, que es mayor a 2.5, por lo tanto seguimos hacia la rama derecha.
3. **Próximo nodo:**
   i. Acá se evalúa si la casa está en la ciudad o en las afueras.
   ii. Como está fuera de la capital, seguimos la rama izquierda (Ubicación ≤ 0.5[41]).
4. **Resultado:**
   i. Al final, llegamos a una hoja con un valor estimado de 320 mil dólares.
   ii. Este es el valor promedio predicho para casas con estas características (menores a 137.5 m², con más de 2.5 habitaciones, fuera de la ciudad[42]).

Así es como el árbol de decisión utiliza características clave como el tamaño, las habitaciones y la ubicación para predecir el precio de la casa en función de los datos proporcionados.

¿Cómo se *entrena* un árbol de decisión?

El entrenamiento de un árbol de decisión consiste en encontrar automáticamente las mejores preguntas (o "divisiones") que puede hacer el árbol en cada nivel mediante un proceso recursivo que optimiza métricas de "impureza" como el índice Gini o la entropía[43]. El algoritmo analiza todas las características disponibles (como metros cuadrados, número de habitaciones, ubicación, etc.) y determina cuál de ellas, y en qué punto específico, maximiza la ganancia de información, es decir, divide mejor los datos en grupos más "puros" o homogéneos. Por ejemplo, el algoritmo podría probar diferentes valores de metros cuadrados (¿100m²? ¿120m²? ¿137.5m²?) y elegir aquel que mejor separa las casas en grupos de precios similares. Este proceso se repite recursivamente en cada nodo hijo hasta alcanzar criterios de parada predefinidos, como una profundidad máxima del árbol, un número mínimo de muestras por hoja, o una ganancia mínima de información por división. El desafío principal está en encontrar el balance óptimo en la complejidad del árbol: si permitimos que el árbol crezca demasiado, podría sobreajustarse a los datos de entrenamiento capturando ruido; si lo limitamos demasiado, podría no capturar patrones importantes (subajuste). Este balance se puede controlar mediante técnicas de "poda" (pruning) y diversos hiperparámetros que regulan el crecimiento del árbol.

En la práctica, los árboles de decisión son especialmente valorados en situaciones donde la interpretabilidad del modelo es tan importante como su precisión. Son ampliamente utilizados en medicina para protocolos de diagnóstico, en finanzas para evaluación de riesgos, y en sistemas de

[41] El árbol de decisión convierte la decisión binaria (capital o fuera) en 1 o 0. Si la ubicación es <= 0.5 eso quiere decir que es 0, en cambio si es > 0.5 será igual a 1.

[42] Asumimos que en este modelo no es tan importante estar cerca de una estación de subte/metro o tener o no un Garage (si quisiera demostrarlo el árbol de decisión no entraría en la página ^_^)

[43] La entropía es una medida del desorden o la incertidumbre en un sistema. En el contexto del machine learning, la entropía se utiliza para medir la impureza o heterogeneidad en un conjunto de datos - cuanto mayor sea la entropía, más desordenados o mezclados están los datos. Este concepto es fundamental en algoritmos de árboles de decisión y en teoría de la información. Si te interesa profundizar en este concepto, puedes encontrar mi artículo detallado "Understanding Entropy made me a better data scientist" en Medium. https://medium.com/@gabrielpierobon/understanding-entropy-made-me-a-better-data-scientist-3196a3ff6ab4

recomendación para categorizar preferencias de usuarios. Sin embargo, rara vez se utilizan de forma aislada; más bien, suelen ser la base para construir modelos más sofisticados como Random Forests o Gradient Boosting, que combinan múltiples árboles para mejorar la precisión mientras mantienen muchas de las ventajas de interpretabilidad. Los científicos de datos solemos comenzar con árboles de decisión simples para entender las relaciones en sus datos antes de avanzar hacia estos modelos más complejos, y frecuentemente los utilizamos para explicar de manera simplificada el comportamiento de otros modelos más complejos a stakeholders no técnicos.

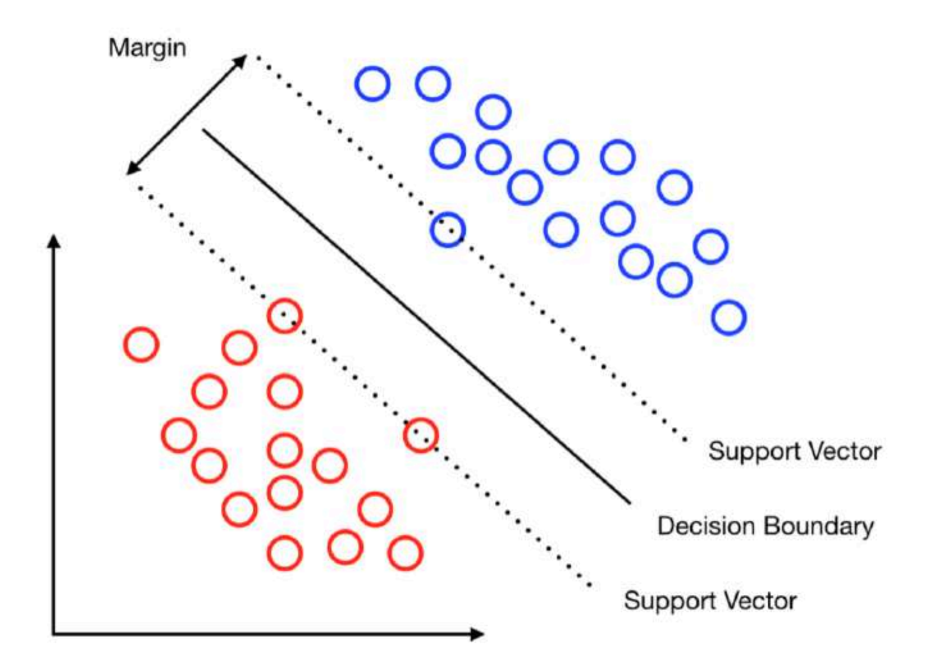
- **Máquinas de Vectores de Soporte (SVM, Support Vectors Machine)**: son algoritmos que buscan encontrar la mejor manera de separar diferentes grupos de datos, como si dibujáramos una línea o frontera entre ellos.

  Imaginá que tenemos puntos de dos colores diferentes en un papel y queremos dibujar una línea que los separe lo mejor posible:

  - El objetivo principal es encontrar la "mejor línea divisora" que mantenga la máxima distancia posible con los puntos más cercanos de cada grupo
  - Los puntos más cercanos a esta línea divisora se llaman "vectores de soporte" porque son los que realmente importan para determinar dónde trazar la frontera
  - Cuando los datos no se pueden separar con una línea recta, SVM usa una técnica especial (el truco del kernel) que es como "doblar el papel" para poder encontrar una mejor separación

  Por ejemplo:

  - 🔍 En reconocimiento facial: puede separar las características que distinguen diferentes rostros
  - 🏥 En diagnóstico médico: puede separar patrones de datos que indican presencia o ausencia de una enfermedad
  - 📧 En filtros de spam: puede distinguir entre patrones de correos legítimos y no deseados



[44]

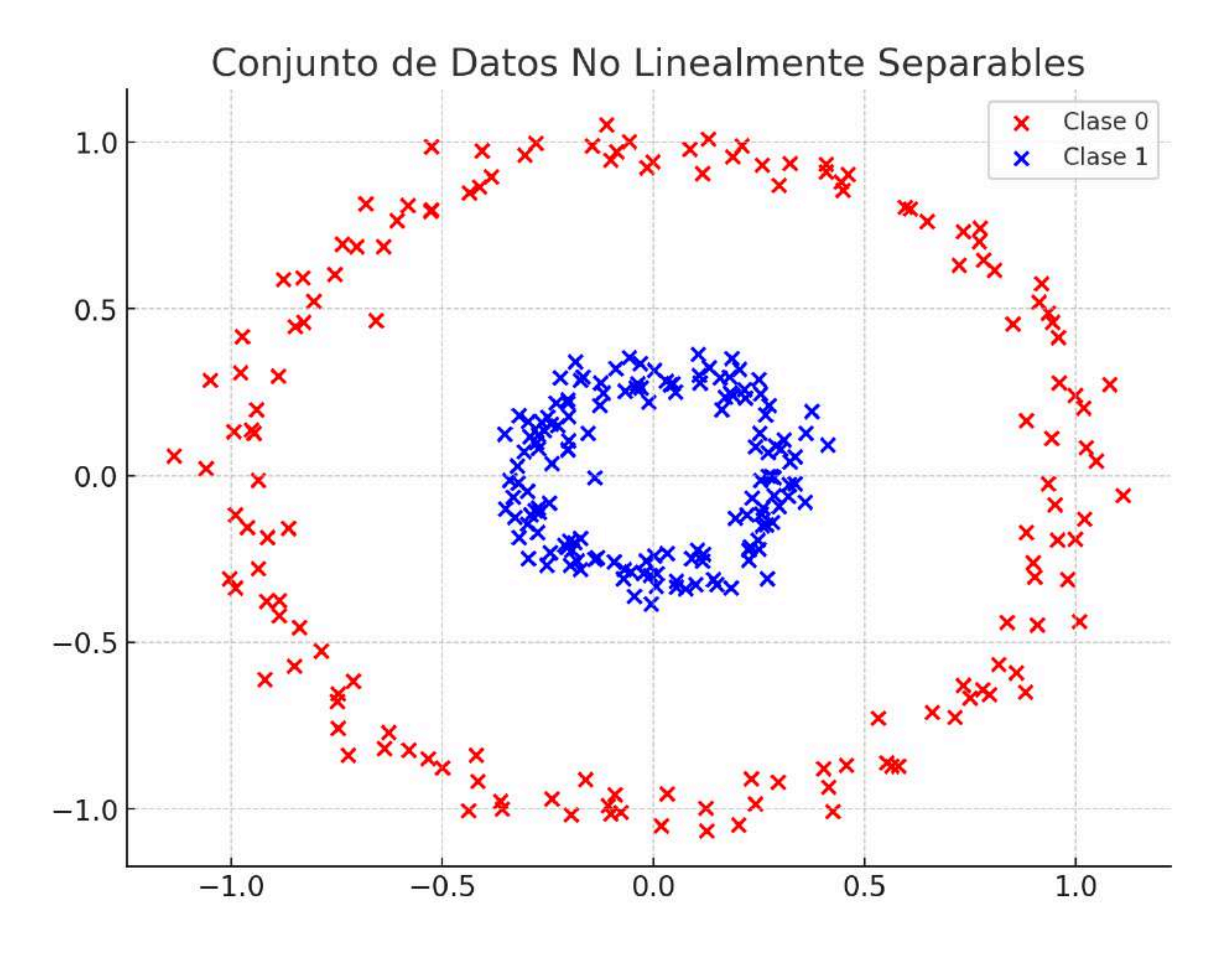
[44] Imágen obtenida en: https://www.iunera.com/kraken/fabric/support-vector-machines-svm/

En el gráfico de la página anterior podemos ver una representación gráfica de una Máquina de Vectores de Soporte (SVM) para un problema de clasificación binaria. El gráfico muestra dos clases de datos, representadas por puntos rojos y azules, distribuidos en un espacio bidimensional definido por dos características. La línea sólida negra en el centro es la línea de decisión (o hiperplano) que separa óptimamente las dos clases. Las líneas discontinuas paralelas a la línea de decisión representan los márgenes, que son las distancias máximas entre la línea de decisión y los puntos más cercanos de cada clase. Estos puntos más cercanos, que "tocan" los márgenes, son los vectores de soporte, cruciales para definir el hiperplano óptimo. La SVM busca maximizar este margen para lograr la mejor separación posible entre las clases, lo que se refleja en la clara división entre las regiones sombreadas en azul claro y rosa.

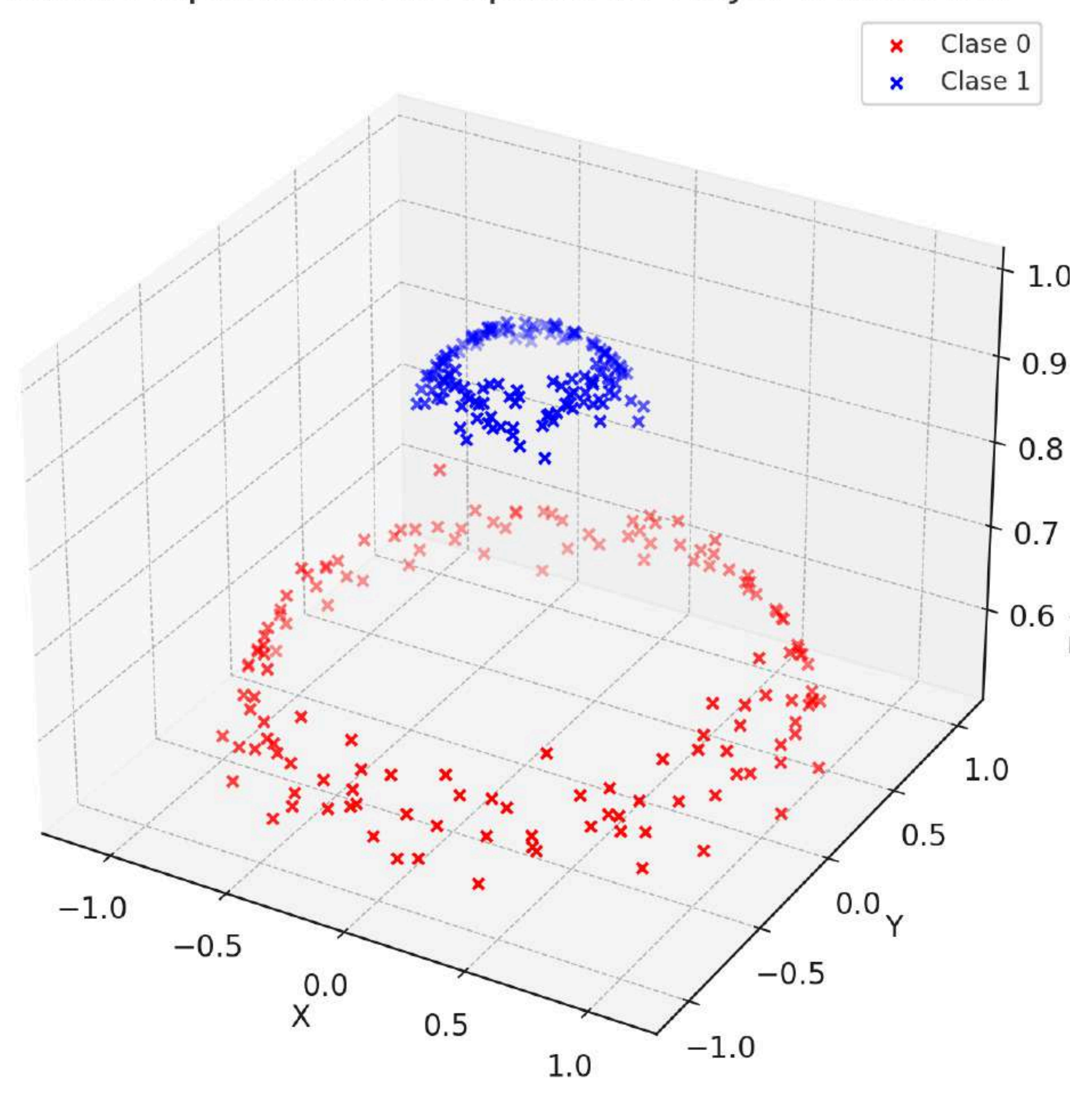
El algoritmo SVM se diferencia de la regresión logística en que, mientras la regresión lineal busca encontrar una línea que mejor se ajuste a todos los puntos minimizando el error total, SVM se centra en encontrar la frontera óptima que maximice la distancia con los puntos más cercanos de cada clase, ignorando los puntos más alejados. En otras palabras, mientras la regresión lineal se preocupa por todos los puntos y trata de acercarse lo más posible a cada uno de ellos, SVM solo se preocupa por los puntos más cercanos a la frontera (los vectores de soporte) y busca mantener la máxima distancia posible con estos, lo que suele resultar en una mejor capacidad para generalizar y clasificar nuevos datos correctamente.

La gran ventaja de SVM, además, es su capacidad para manejar problemas complejos donde la separación entre grupos no es obvia a simple vista. Lo que me interesa que entiendas de SVM es que si bien es principalmente un modelo lineal como la regresión lineal/logística, además utiliza un *truco* adicional para tratar con datos donde una línea recta *no podría pasar.* Este truco es el *truco del kernel*, que utiliza transformaciones matemáticas para *contorsionar* los datos de una forma en la cual la línea recta pueda pasar.

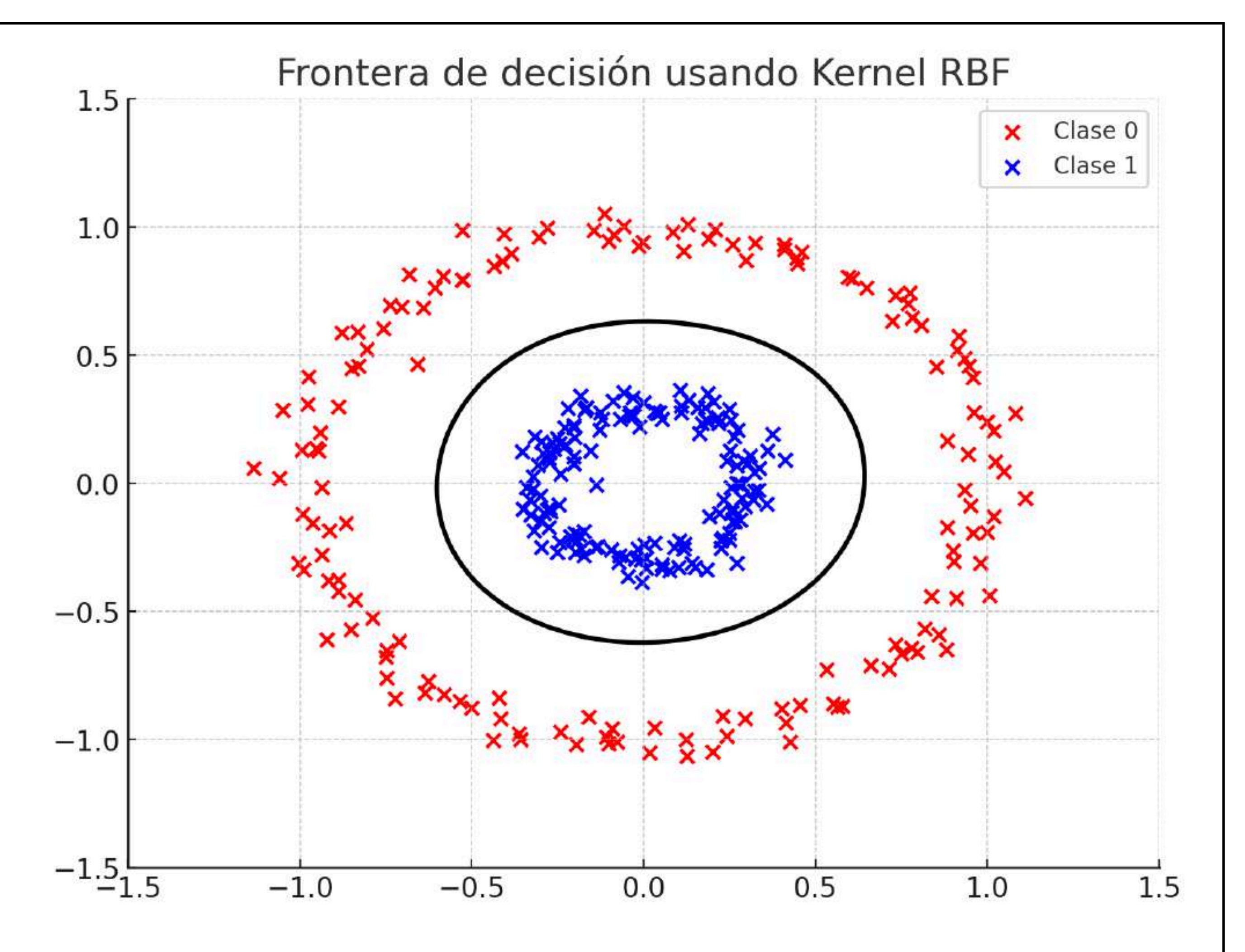
Imaginate que tenemos unos datos con esta estructura:



En esta dimensión y bajo estas circunstancias no podemos usar un simple modelo lineal para separar ambas clases. Pero imaginate que pudiéramos entonces proyectar estos datos en una nueva dimensión. Pensemos simplemente en que pudiera haber una tercera dimensión para estos datos.



¿Podés ver cómo ahora esos datos sí son separables por un *plano* que pase entre los puntos rojos y los azules? Tiene que ser un plano (como una hoja de papel) y no una línea por que estamos en tres dimensiones, pero eso para nosotros no es un problema, porque ese plano para nosotros sería a todos los efectos ese modelo lineal que estábamos buscando. Entonces al utilizar este *truco* de transformar los datos a una representación dimensional diferente, encontramos el plano que buscábamos y somos capaces de separar las clases de la siguiente manera:



El circulo negro es la proyección del plano *cortando* el espacio entre los puntos rojos y los azules y para nosotros se convierte en un modelo que separa perfectamente ambas clases.

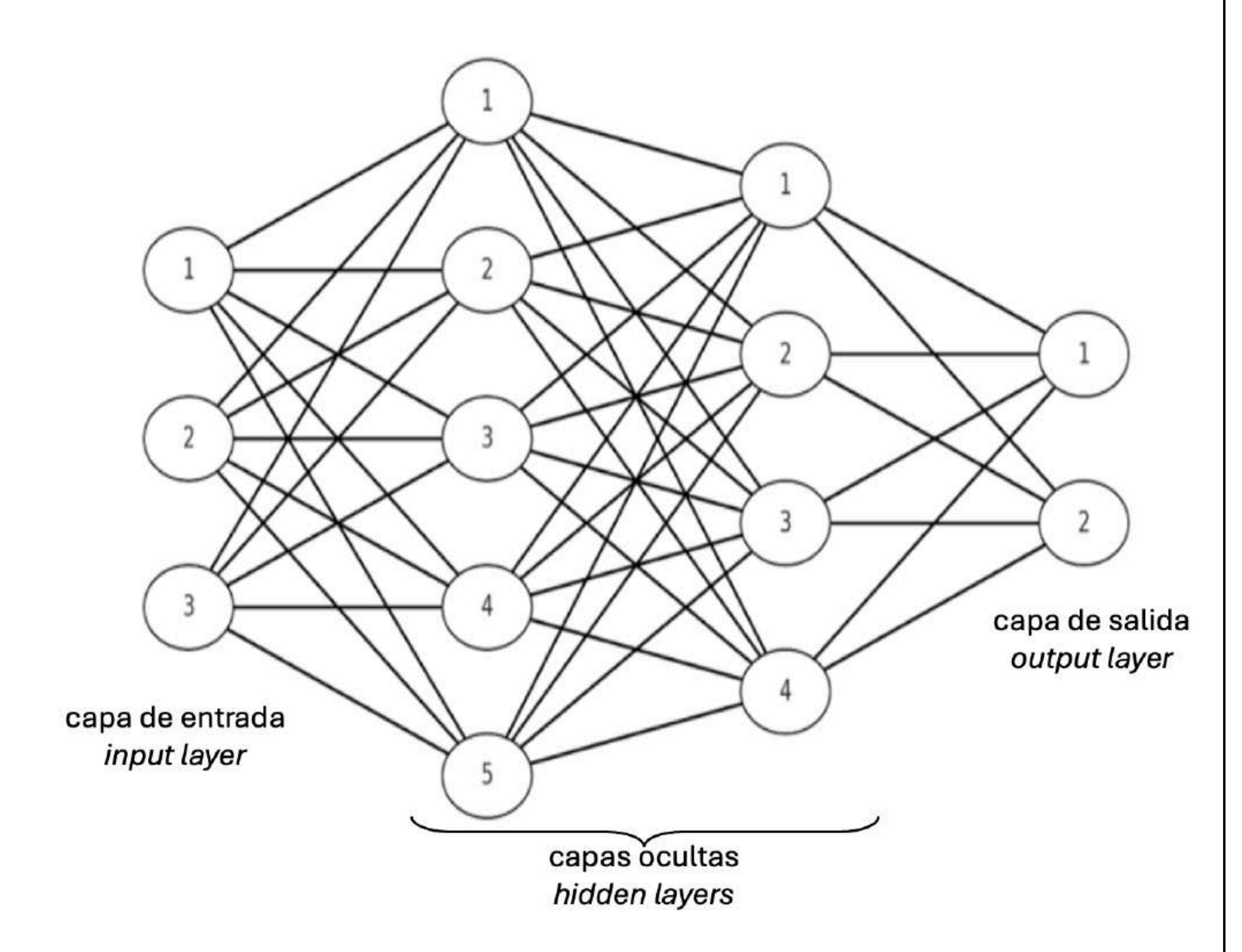
¡Esto es lo que un algoritmo como SVM permite hacer!

¿Potente no?

En la práctica, SVM sigue siendo una herramienta muy valorada en el arsenal de un científico de datos, especialmente en situaciones donde los datos son complejos pero el tamaño del conjunto de datos es moderado. Es particularmente efectivo en problemas de clasificación de texto, análisis de imágenes y en campos como la bioinformática, donde la dimensionalidad de los datos es alta pero el número de muestras puede ser limitado. Su capacidad para manejar relaciones no lineales a través del truco del kernel, combinada con su robustez ante el ruido en los datos, lo hace especialmente útil en aplicaciones del mundo real. Sin embargo, con el surgimiento de técnicas más avanzadas como las redes neuronales, que veremos a continuación, SVM cedió terreno en algunas aplicaciones como el reconocimiento de imágenes a gran escala. No obstante, mantiene su relevancia en problemas donde la interpretabilidad y la eficiencia computacional son prioritarias, o cuando no se dispone de los grandes volúmenes de datos que las redes neuronales típicamente requieren para su entrenamiento.

- **Redes Neuronales**: las famosísimas redes neuronales son modelos que están inspirados en el funcionamiento del cerebro humano. Consisten en capas interconectadas de neuronas artificiales que procesan y transmiten información. Estas redes pueden encontrar patrones complejos y no lineales en los datos, lo que las hace extremadamente versátiles. Se componen típicamente de una

capa de entrada (*input layer*), una o más capas ocultas (*hidden layers*), y una capa de salida (*output layer*). Cada neurona recibe inputs, los procesa mediante funciones de activación, y pasa el resultado a la siguiente capa.



Esta imagen muestra la estructura de una red neuronal artificial estándar. Está formada por cuatro capas: una de entrada con 3 neuronas, dos capas ocultas con 5 y 4 neuronas respectivamente, y una capa de salida con 2 neuronas. Cada neurona, representada por un círculo numerado, está conectada a todas las neuronas de la capa siguiente mediante líneas que representan las conexiones de la red. Esta configuración permite que la información fluya de izquierda a derecha, procesándose a través de las capas para transformar los datos de entrada en una salida deseada, capturando patrones complejos en el proceso. En su propia sección, veremos las redes neuronales en detalle.

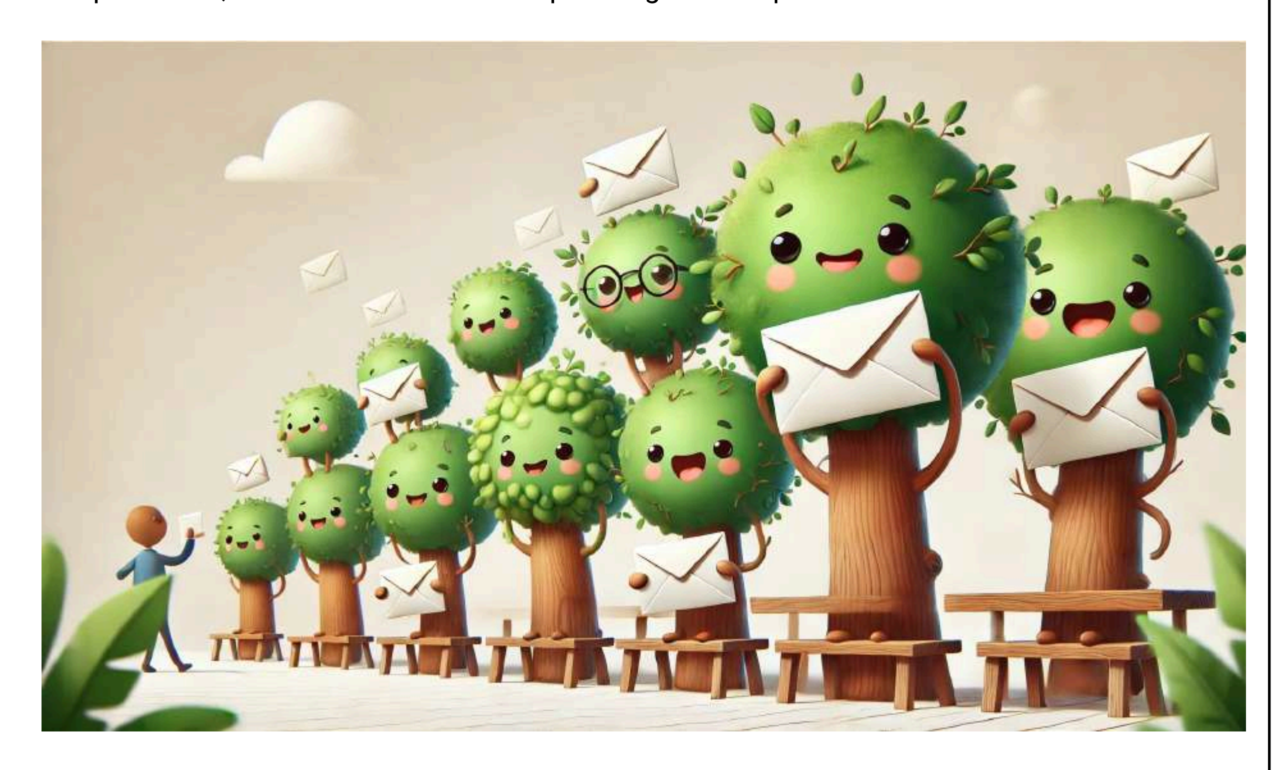
Tenemos una sección completa dedicada a Redes Neuronales y Deep Learning más adelante así que por ahora lo dejamos acá.

- ***Ensemble Methods***: los métodos de ensamble (*Ensemble Methods*) se basan en un principio similar al de la "sabiduría de la multitud" (*wisdom of the crowd*): la idea de que un grupo diverso de modelos puede tomar mejores decisiones que un modelo individual

  Esto se puede entender con una analogía común: cuando buscamos un diagnóstico médico importante, a menudo consultamos a varios doctores. Cada doctor puede tener sus sesgos o especializaciones, pero al combinar múltiples opiniones médicas, tendemos a obtener una visión más completa y precisa. Los métodos de ensamble funcionan de manera similar:

  - **Bagging** (como *Random Forest*): Es como pedir a varios médicos que hagan su diagnóstico

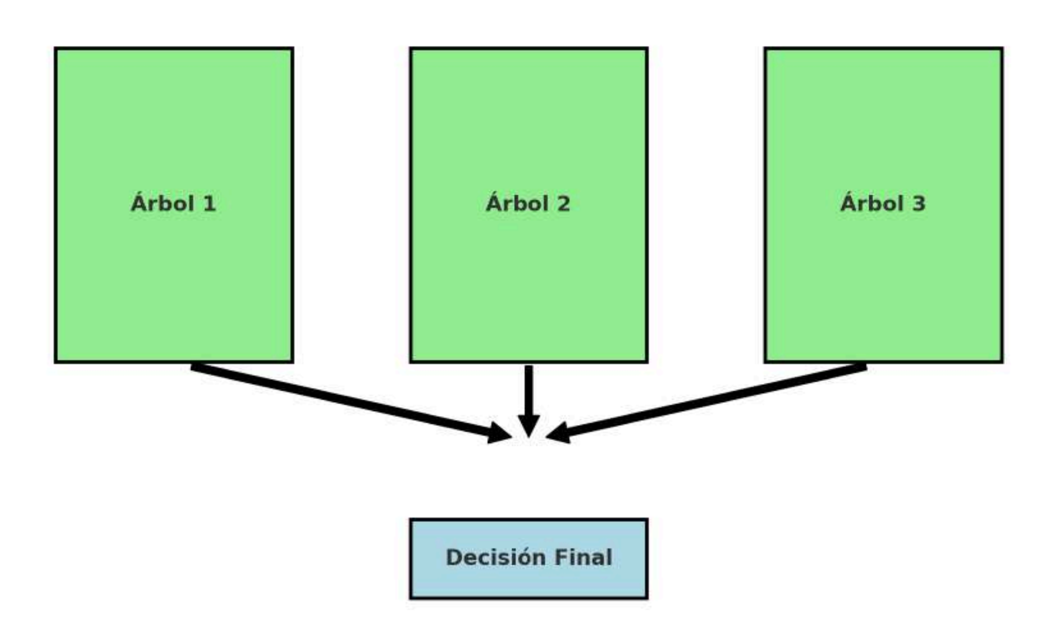
de forma independiente y luego tomar una decisión por votación mayoritaria. **Random Forest** es un método de ensamble que construye múltiples árboles de decisión de forma paralela e independiente, donde cada árbol "vota" para llegar a una predicción final.



Podemos imaginarlo como reunir a un grupo diverso de expertos, donde cada uno (un árbol) analiza el problema con una perspectiva ligeramente diferente porque:

- Cada experto solo tiene acceso a una parte aleatoria de la información disponible
- Cada uno considera un subconjunto diferente de características para tomar sus decisiones.
- Por ejemplo, en la predicción del precio de una casa, un árbol podría enfocarse más en el tamaño y la ubicación, otro en la antigüedad y el número de habitaciones, y así sucesivamente.
- La decisión final se toma "democráticamente": en clasificación por mayoría de votos, y en regresión promediando las predicciones de todos los árboles.
- Esta diversidad hace que Random Forest sea más robusto y menos propenso a errores que un único árbol de decisión.

En la práctica, Random Forest se ha convertido en uno de los algoritmos más utilizados en la industria por su versatilidad y robustez. Es especialmente valorado porque requiere poca preparación de los datos (puede manejar valores faltantes y diferentes tipos de variables), no necesita una extensa calibración de parámetros para obtener buenos resultados, y es menos propenso al sobreajuste que otros modelos. Se utiliza extensivamente en aplicaciones como sistemas de recomendación, detección de fraude bancario, predicción de fallas en equipos industriales, y análisis de riesgo crediticio. Además, proporciona de forma natural medidas de la importancia de cada variable en las predicciones, lo que lo hace particularmente útil en etapas de análisis exploratorio y selección de características. Sin embargo, cuando los datos tienen patrones muy complejos o estructura temporal, otros modelos como las redes neuronales o el gradient boosting podrían ser más apropiados.



Esta imagen ilustra de manera simplificada el funcionamiento de un modelo Random Forest, un ejemplo clásico de método ensemble. Hay tres rectángulos verdes, cada uno representando un árbol de decisión independiente (Árbol 1, Árbol 2 y Árbol 3). Estos árboles están conectados a un rectángulo azul en la parte inferior etiquetado como "Decisión Final". Esta estructura visual representa cómo Random Forest combina las predicciones de múltiples árboles de decisión para llegar a una decisión final más robusta y precisa. Cada árbol aporta su predicción basada en un subconjunto diferente de datos o características, y el modelo final agrega estas predicciones individuales para generar un resultado consensuado, lo que típicamente resulta en un mejor rendimiento que el de cualquier árbol individual.

- **Boosting**: Es como tener una serie de médicos donde cada uno se especializa en los casos que los médicos anteriores encontraron más difíciles. El **Gradient Boosting** es un método de ensamble que construye modelos de manera secuencial, donde cada nuevo modelo se enfoca en corregir los errores de los modelos anteriores. Podemos entenderlo como un proceso de aprendizaje iterativo donde cada nuevo modelo es un "experto" en corregir los errores que persistían hasta ese momento. Por ejemplo, si estamos prediciendo precios de casas, el primer modelo podría dar una estimación básica, el segundo se especializaría en corregir las predicciones que quedaron muy altas, el tercero en las que quedaron muy bajas, y así sucesivamente. Cada modelo nuevo presta especial atención a los casos donde los modelos anteriores tuvieron mayor error, similar a cómo un estudiante se enfoca en los temas donde ha cometido más errores. La predicción final combina todas estas correcciones, resultando en un modelo más preciso y robusto.

  En la práctica, el Gradient Boosting, especialmente a través de implementaciones populares como XGBoost, LightGBM y CatBoost, se convirtió en uno de los algoritmos dominantes en competencias de ciencia de datos y aplicaciones del mundo real. Es particularmente efectivo en problemas donde se requiere la máxima precisión posible, como predicción de comportamiento de usuarios, pronósticos financieros, y sistemas de detección de riesgos. A diferencia de Random Forest, requiere una calibración más cuidadosa de sus *hiperparámetros* para evitar el sobreajuste, pero cuando está bien ajustado, suele superar a otros algoritmos en términos de precisión. Su capacidad para manejar naturalmente diferentes tipos de datos, combinada con implementaciones altamente optimizadas que permiten procesar grandes volúmenes de información de manera eficiente, lo convirtieron en la herramienta preferida en

muchas aplicaciones.

Vemos entonces que tanto en *Bagging* como en *Boosting* prima el concepto de que, al igual que una multitud diversa suele estimar mejor que un solo experto (por ejemplo, al adivinar la altura de una persona o el número de caramelos en un frasco[45]), un conjunto de modelos diversos suele predecir mejor que un modelo individual, compensando los errores y sesgos de cada uno.











Ventajas del aprendizaje supervisado

- **Precisión**: Puede alcanzar niveles muy altos de precisión si se dispone de suficientes datos etiquetados y se selecciona el modelo adecuado. Imaginá enseñarle a un niño a diferenciar perros de gatos mostrándole muchas fotos con las respuestas correctas - cuantos más ejemplos vea, mejor será su capacidad para distinguirlos. De manera similar, cuando alimentamos un modelo con datos etiquetados de calidad, este puede aprender patrones muy específicos y realizar predicciones con gran exactitud.
- **Explicabilidad del modelo**: Muchos algoritmos de aprendizaje supervisado permiten interpretar cómo se toman las decisiones, lo que es crucial en aplicaciones donde la explicabilidad es importante. Por ejemplo, en un árbol de decisión para aprobar préstamos bancarios, podemos ver exactamente qué factores (ingresos, historial crediticio, etc.) llevaron a una decisión específica. Esta

[45] Este fenómeno, conocido como "sabiduría de las multitudes", fue observado por Francis Galton en 1907 en una feria de ganado, donde el promedio de las estimaciones de 787 personas sobre el peso de un buey fue sorprendentemente preciso, con un error de solo 1% respecto al peso real. Es algo que podemos observar en nuestra vida cotidiana: cuando un grupo diverso de personas intenta adivinar algo (el peso de una valija, la edad de alguien, o la duración de un trayecto), algunos estimarán por encima y otros por debajo, pero el promedio de todas las estimaciones suele ser notablemente preciso. Este principio ilustra cómo los errores individuales tienden a cancelarse entre sí cuando se promedian múltiples estimaciones independientes.

transparencia es fundamental en sectores regulados como finanzas o medicina, donde necesitamos entender y justificar las decisiones del modelo.
- **Aplicabilidad**: Es ampliamente aplicable en diversas industrias y tipos de datos, desde texto, imágenes hasta series temporales y datos estructurados. Por ejemplo, puede usarse para predecir el mantenimiento necesario en maquinaria industrial analizando datos de sensores (lo vamos a ver más adelante en este libro), para detectar fraudes en transacciones bancarias (también lo vamos a ver en detalle), o para clasificar correos electrónicos como spam. Esta versatilidad lo hace valioso en casi cualquier campo donde existan datos etiquetados.

Desafíos del aprendizaje supervisado

- **Necesidad de Datos Etiquetados**: Necesita grandes cantidades de datos etiquetados, lo que puede ser costoso y llevar mucho tiempo obtener. Pensá en querer crear un modelo para diagnosticar enfermedades raras - necesitarías miles de imágenes médicas correctamente diagnosticadas por especialistas, lo cual es costoso y consume mucho tiempo. En algunos campos, obtener expertos para etiquetar datos puede ser tan caro que hace inviable el proyecto. Además, el proceso de etiquetado debe ser consistente y preciso, ya que errores en las etiquetas pueden llevar a un modelo mal entrenado.
- ***Overfitting***: Como ya mencioné antes, si un modelo es demasiado complejo, puede ajustarse demasiado bien a los datos de entrenamiento y no generalizar bien a nuevos datos. Imaginá a un estudiante preparándose para un examen de matemáticas que, en lugar de entender los conceptos y cómo resolver diferentes tipos de problemas, simplemente memoriza todas las soluciones exactas de los ejercicios del libro de práctica, incluyendo cada número específico. Durante el "entrenamiento" (estudiando los ejercicios del libro), su rendimiento es perfecto porque las memorizó exactamente. Sin embargo, durante el "test" (el examen real), cuando los problemas son nuevos y tienen números diferentes, el estudiante falla porque no aprendió el patrón general de cómo resolver problemas matemáticos, sino que solo memorizó casos específicos. Al igual que este estudiante, un modelo con overfitting puede tener un rendimiento excepcional en los datos con los que fue entrenado, pero fallar significativamente cuando se enfrenta a nuevos casos.
- **Sesgo y Varianza**: El sesgo y la varianza son dos tipos de errores que necesitamos balancear cuando construimos un modelo de machine learning. El equilibrio entre sesgo y varianza es fundamental porque representa un compromiso inevitable en el aprendizaje automático: los modelos con alto sesgo tienden a ser demasiado simples y pueden perder patrones importantes en los datos (como una línea recta tratando de ajustar una relación claramente curva), mientras que los modelos con alta varianza son demasiado complejos y sensibles a pequeños cambios en los datos de entrenamiento (como un polinomio de alto grado que pasa exactamente por todos los puntos pero oscila salvajemente entre ellos). Por eso, encontrar el punto óptimo entre estos dos extremos es crucial para desarrollar modelos que sean tanto precisos como robustos. Podemos entenderlo mejor con una analogía de tiro al blanco (ilustrado en la siguiente página):
    - **Sesgo (Bias)**: Es como disparar sistemáticamente al mismo lugar, pero lejos del centro. Un modelo con alto sesgo es demasiado simple y comete errores consistentes:
        - Por ejemplo, si intentamos predecir el precio de casas solo mirando su tamaño, ignorando factores importantes como la ubicación, estaremos cometiendo errores sistemáticos
        - Este problema se llama "underfitting" (sub-ajuste): el modelo es tan simple que no puede capturar los patrones importantes en los datos
    - **Varianza**: Es como disparar de manera dispersa, a veces cerca y a veces lejos del centro. Un modelo con alta varianza es demasiado complejo y sensible a pequeños cambios en los datos. Por ejemplo, si nuestro modelo de precios de casas considera hasta el más mínimo detalle (como el color de las paredes o el número de ventanas), puede funcionar muy bien con los datos de entrenamiento pero fallar con casos nuevos

El arte está en encontrar el equilibrio: queremos un modelo que sea lo suficientemente complejo para capturar los patrones importantes, pero lo suficientemente simple para generalizar bien a nuevos casos.

En la práctica, este equilibrio se logra principalmente a través de la validación cruzada y la experimentación sistemática con diferentes configuraciones del modelo. Los científicos de datos típicamente dividen sus datos en tres conjuntos (entrenamiento, validación y prueba) y prueban diferentes niveles de complejidad del modelo, observando cómo el rendimiento en el conjunto de validación cambia a medida que se ajustan los hiperparámetros que controlan la complejidad del modelo (como la profundidad máxima en árboles de decisión, el número de capas en redes neuronales, o los parámetros de regularización en modelos lineales). La configuración que obtiene el mejor rendimiento en el conjunto de validación suele ser la que proporciona el mejor equilibrio entre sesgo y varianza.

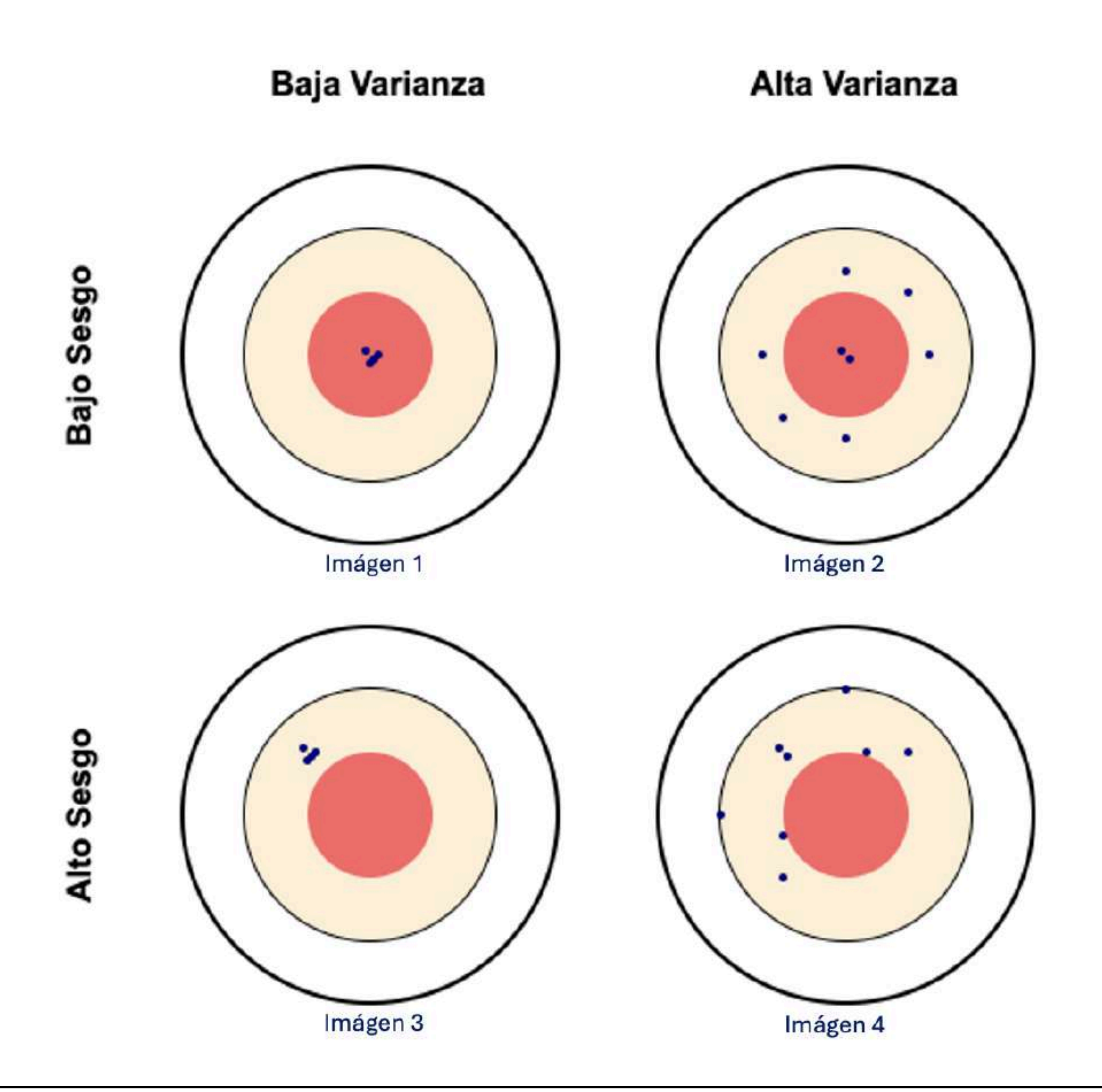


Imagen 1 (Bajo Sesgo, Baja Varianza): *El modelo tiene una alta precisión y es consistente en sus predicciones. Los puntos están agrupados cerca del centro, lo que indica que el error sistemático (sesgo) es bajo y que las predicciones son poco dispersas (baja varianza).*

Imagen 2 (Bajo Sesgo, Alta Varianza): *El modelo tiene una buena precisión en promedio, pero las predicciones son inconsistentes. Los puntos están dispersos alrededor del centro, lo que sugiere un sesgo bajo pero una varianza alta.*

Imagen 3 (Alto Sesgo, Baja Varianza): *El modelo es consistente, pero tiene un error sistemático significativo. Los puntos están agrupados lejos del centro, lo que indica un sesgo alto, aunque la varianza sigue siendo baja.*

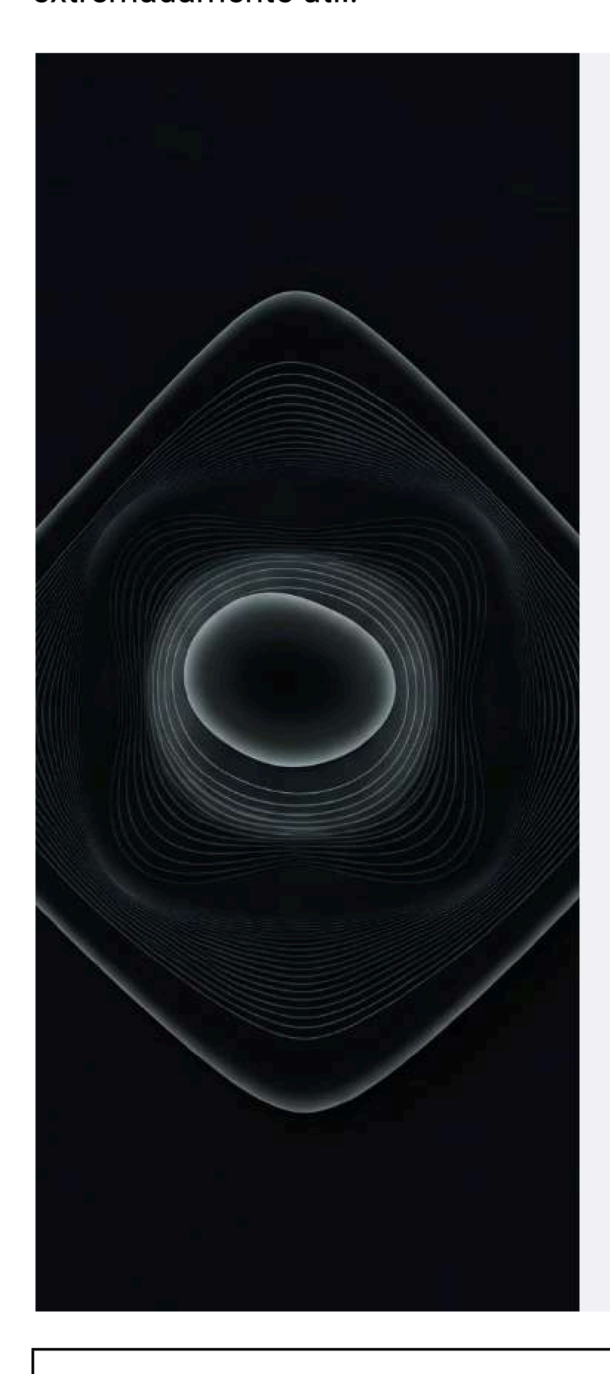
Imagen 4 (Alto Sesgo, Alta Varianza)*: El modelo tiene un error sistemático elevado y es inconsistente en sus predicciones. Los puntos están dispersos lejos del centro, lo que indica tanto un sesgo alto como una alta varianza.*

Llegamos al final de nuestra larga sección sobre aprendizaje supervisado y es momento de preguntarnos ¿qué podemos hacer cuando los datos que tenemos no están *etiquetados*? ¿Qué clase de análisis podemos hacer sobre un conjunto de datos de esa naturaleza? ¿Es posible detectar *fraude* si no tenemos datos que han sido identificados como tal previamente?

Es momento de que hablemos de *aprendizaje NO supervisado*.

04. 🔍 Aprendizaje No Supervisado (Machine Learning)

El aprendizaje *no supervisado* permite que los modelos encuentren patrones y estructuras en datos *no etiquetados*. Es fundamental para tareas como el *clustering* y la *reducción de dimensionalidad*, que son muy importantes para la exploración y análisis de grandes volúmenes de datos. Es de vital importancia además porque en la práctica termina siendo casi más normal disponer de datos **no** etiquetados para trabajar y es una herramienta que te recomiendo que entiendas muy bien en qué casos usar porque puede resultarte extremadamente útil.



Conceptos Clave del Aprendizaje No Supervisado

1. **Datos No Etiquetados**
   Los datos de entrenamiento no tienen etiquetas asociadas. El objetivo es identificar estructuras y patrones inherentes.
2. **Agrupamiento (Clustering)**
   Agrupar datos en clusters donde los objetos son más similares entre sí que a otros grupos.
3. **Reducción de Dimensionalidad**
   Reducir variables para simplificar modelos y descubrir estructuras subyacentes más relevantes en los datos.
4. **Detección de Anomalías**
   Identificar datos que no se ajustan a un patrón esperado o a un conjunto mayoritario.

A diferencia del aprendizaje *supervisado* que vimos en la sección anterior, donde se utilizan datos etiquetados para entrenar un modelo, el aprendizaje *no supervisado* trabaja con datos sin etiquetas, lo que lo

hace especialmente útil para explorar datos y descubrir relaciones ocultas.

Por ejemplo, en una base de datos de correos electrónicos:

- **Datos etiquetados** serían correos donde ya sabemos si son spam o no: "Gana dinero rápido" → SPAM, "Reunión mañana 10am" → NO SPAM. El modelo aprende de estas etiquetas para clasificar futuros correos.
- **Datos no etiquetados** serían simplemente una colección de correos sin clasificar: el algoritmo podría agruparlos basándose en sus características comunes (longitud, palabras frecuentes, horario de envío) sin saber explícitamente cuáles son spam y cuáles no.

Supongamos que sos el/la director(a) de marketing de una gran empresa de comercio electrónico.



Tenés una enorme base de datos de clientes con información sobre sus compras, navegación en el sitio web y datos demográficos, pero te sentís abrumado(a) por la cantidad de información y no sabés cómo usarla eficazmente. Acá es donde el *clustering*, una técnica de aprendizaje no supervisado, se vuelve tu aliado estratégico. Esta herramienta puede analizar todos esos datos y agrupar automáticamente a tus clientes en segmentos con características similares. Por ejemplo, podrías descubrir un grupo de compradores frecuentes de electrónica de alta gama, otro de padres jóvenes que buscan ofertas en productos para bebés, y un tercero de aficionados a la moda que siguen las últimas tendencias. Sin necesidad de categorías predefinidas (es decir sin necesidad de las *famosas* etiquetas), el clustering te ayuda a entender los patrones naturales en tu base de clientes, permitiéndote crear campañas de marketing más dirigidas, personalizar la experiencia de usuario en tu sitio web y optimizar tu inventario según las preferencias de cada grupo.

Vamos a explorar tres algoritmos clave que te van a permitir llevar esta técnica a la práctica. Primero, vamos a ver *K-means*, un método popular y eficiente que divide tus clientes en un número predefinido de grupos, ideal para cuando tenés una idea aproximada de cuántos segmentos querés crear. Después, nos vamos a

adentrar en el clustering *jerárquico*, que construye una estructura en forma de árbol de tus datos, permitiéndote ver cómo se relacionan los diferentes grupos de clientes entre sí a distintos niveles de similitud. Por último, vamos a explorar *DBSCAN*, un algoritmo robusto que puede detectar grupos de formas irregulares y es excelente para identificar segmentos de clientes poco comunes o nichos de mercado que podrían pasar desapercibidos con otros métodos. Cada uno de estos algoritmos tiene sus fortalezas y se adapta a diferentes escenarios de análisis de clientes, dándote un conjunto de herramientas versátil para sacar el máximo provecho de tus datos.

Recordá estos conceptos:

- **Clustering** es una técnica **no supervisada** para agrupar datos similares automáticamente
- Es útil para descubrir patrones naturales **sin categorías predefinidas (etiquetas)**
- **K-means** es eficiente para crear un número fijo de grupos de clientes
- **Clustering jerárquico** es una estructura en forma de árbol de relaciones entre grupos
- **DBSCAN** detecta grupos irregulares y nichos de mercado ocultos

Ahora sí, vamos a ver estos algoritmos.

Algoritmos de aprendizaje no supervisado

- **K-Means Clustering**: K-Means Clustering es un algoritmo de aprendizaje no supervisado cuyo objetivo es agrupar los datos en *K* clusters basándose en la distancia entre los puntos de datos y los *centroides*[46] de los clusters. El proceso de entrenamiento de este algoritmo comienza inicializando *K* centroides[47] aleatoriamente[48], luego asigna cada punto de dato al centroide más cercano (ilustrativamente podrías pensar como que vamos pintando cada punto de dato con el color del centroide que le cayó más cerca). Luego iterativamente se van recalculando los centroides y reasignando los puntos de datos hasta que las asignaciones se estabilicen, esto es, hasta el punto en el que ya no haya una ganancia por ajustar los centroides. Mejor veamos el proceso de forma ilustrada para que puedas fijar el concepto.

  El funcionamiento de K-Means implica, como ya vimos, cada uno de estos siguientes pasos iterativos:

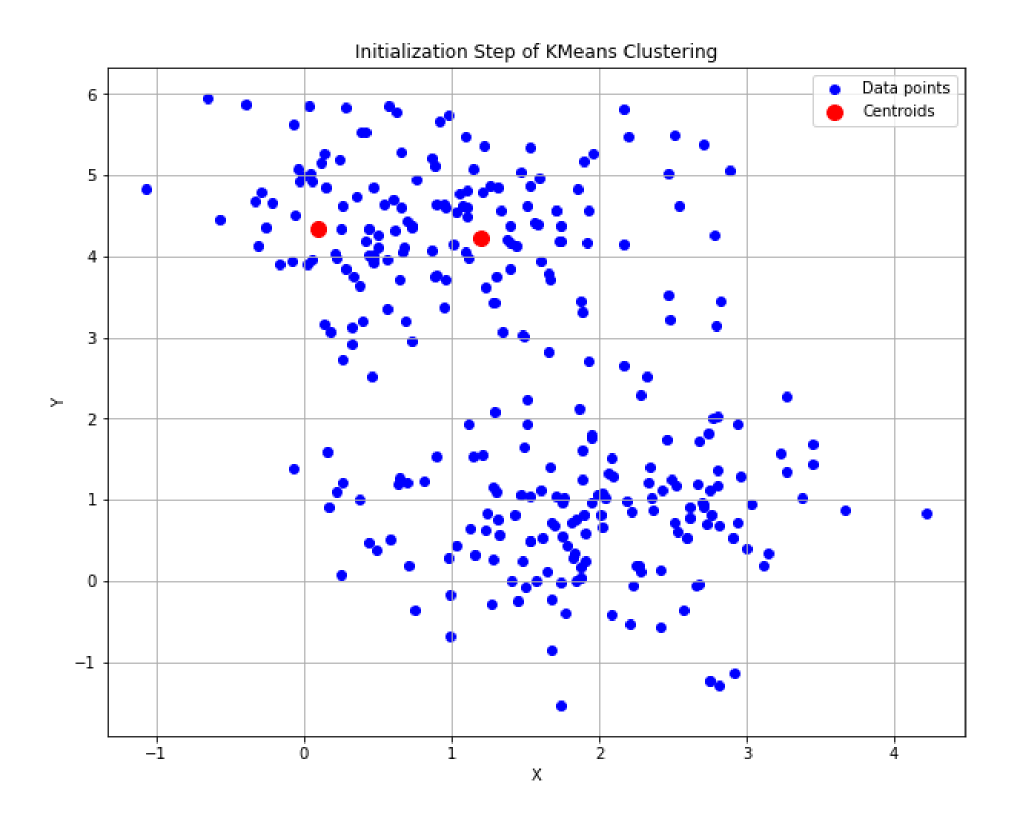
  1. **Inicialización**: Se seleccionan aleatoriamente *K* centroides iniciales para los grupos. Acá podemos ver cómo hemos colocado dos centroides al azar en nuestros datos que aún no han

[46] Un centroide es el punto central representativo de un cluster en algoritmos de agrupamiento como K-Means. Se calcula como el promedio de todos los puntos pertenecientes al cluster y se utiliza para asignar nuevos datos al grupo más cercano. Durante el proceso iterativo del algoritmo, los centroides se actualizan constantemente hasta que se alcanza una convergencia, definiendo así las características finales de cada cluster. Me gusta pensar en el centroide como el *agujero negro* del centro de una galaxia, donde el resto de estrellas y planetas son los datos.

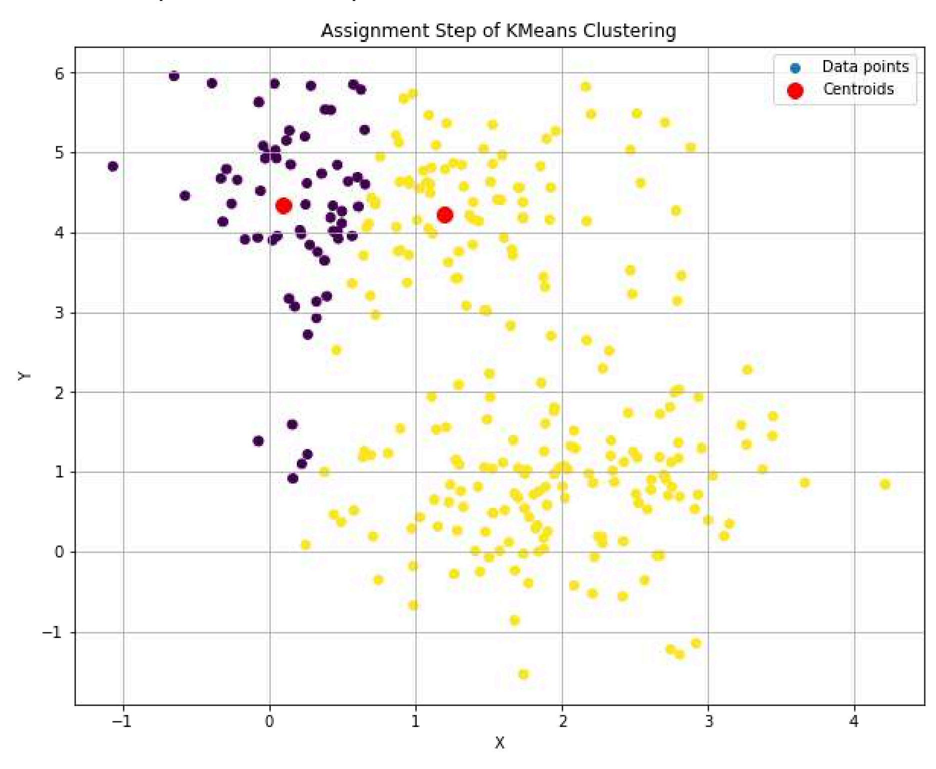
[47] *K* es un número entero que puede ser definido subjetivamente por quienes conducen el experimento o puede ser optimizado mediante técnicas. Es común en la práctica tener una idea previa de cuántos segmentos estamos buscando, con lo cual podríamos comenzar siguiendo nuestra intuición y ver qué resultados obtenemos inicialmente y luego probar subiendo o bajando K. Aunque como ya comenté, se puede utilizar un método para encontrar un K óptimo.

[48] Por *aleatoriamente* nos referimos a que da igual dónde coloquemos estos centroides inicialmente, el algoritmo convergerá a los centroides óptimos.

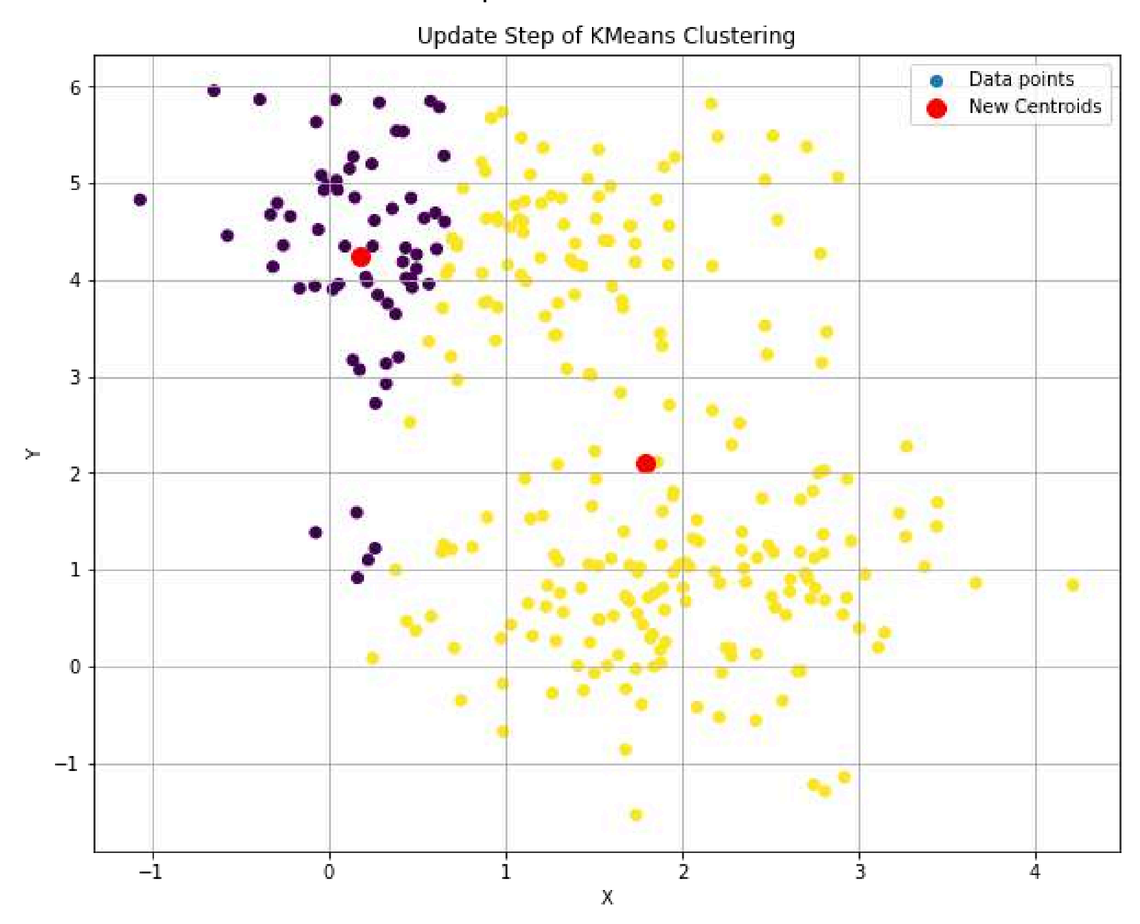
sido segmentados (es decir, aún son todos puntos de datos *azules*).



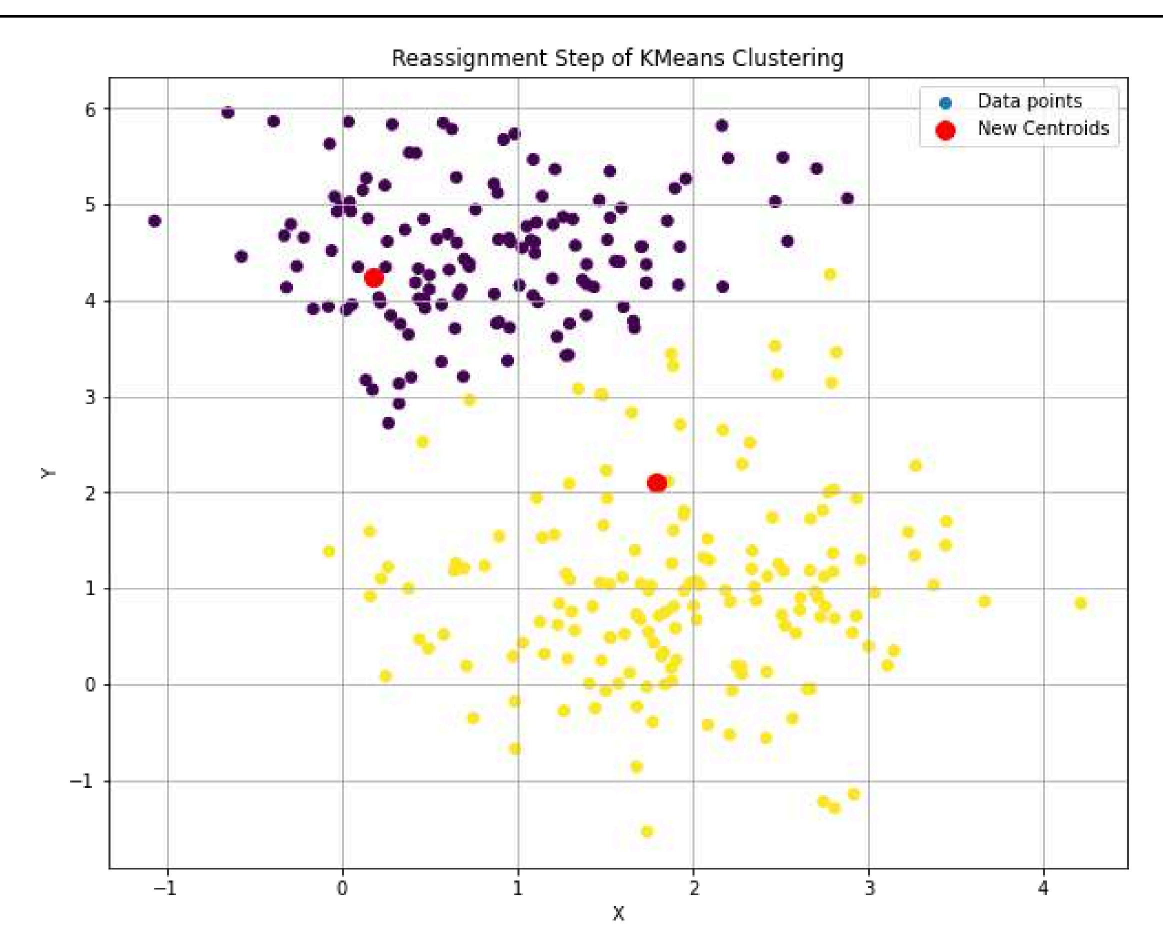
2. **Asignación**: Se asigna cada punto del conjunto de datos al centroide más cercano, formando *K* grupos (en este caso *K es 2* y por lo tanto *pintamos* 2 segmentos). La medida de "cercanía" es típicamente la distancia euclidiana, aunque también se pueden usar otras medidas de distancia. El color que se da a los puntos es entonces el del centroide más cercano.



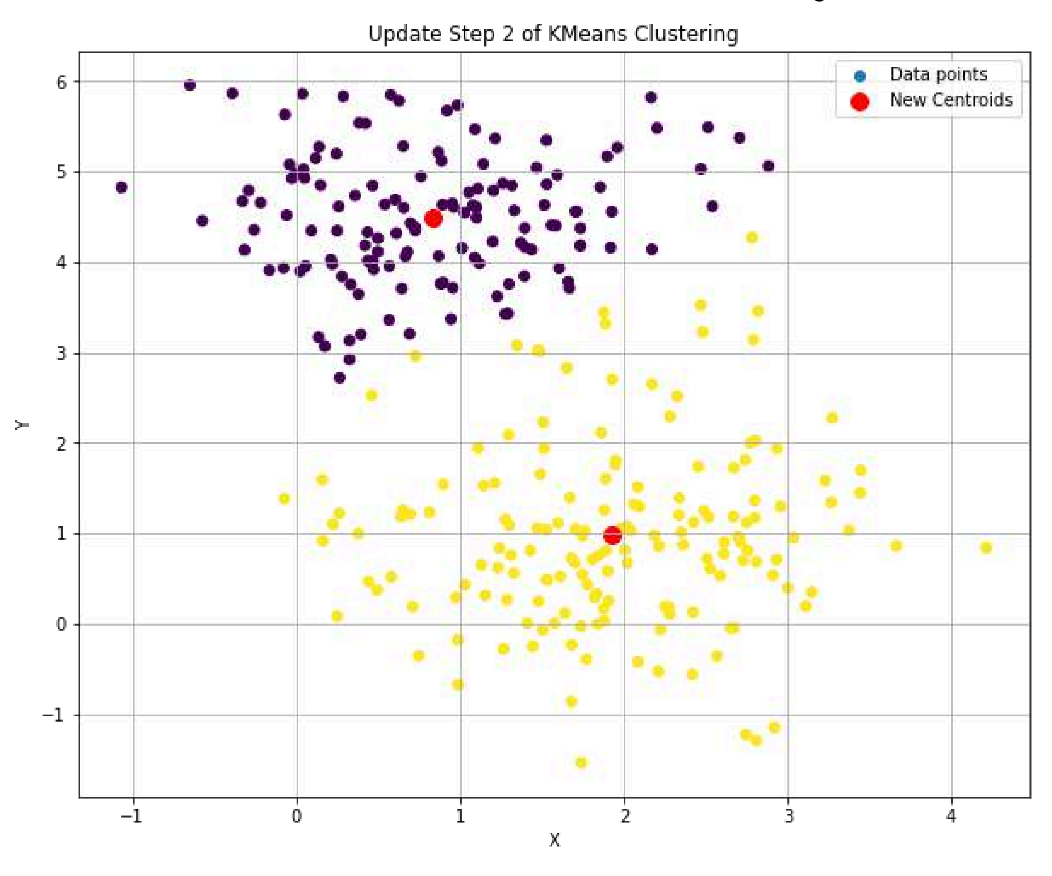
3. **Actualización**: Para cada uno de los *K* grupos, se calcula el *nuevo centroide* como la media de todos los puntos de datos asignados a ese grupo. Podemos ver entonces como cada punto rojo (centroide) se ubica en el centro de los datos que le fueron asignados momentáneamente. Fijate que si bien el centroide del cluster violeta casi no se movió, el centroide del cluster amarillo se desplazó bastante.



4. **Repetición**: Repetir los pasos 2 y 3 hasta que se cumpla un criterio de finalización. Esto podría ser un número determinado de iteraciones, o cuando la asignación de puntos de datos a los grupos ya no cambia (se dice que el algoritmo *ha convergido*). Es decir que luego del punto 3, otra vez podemos volver a *pintar* los datos según su cercanía con la nueva ubicación de los centroides. Llegaríamos a algo así:

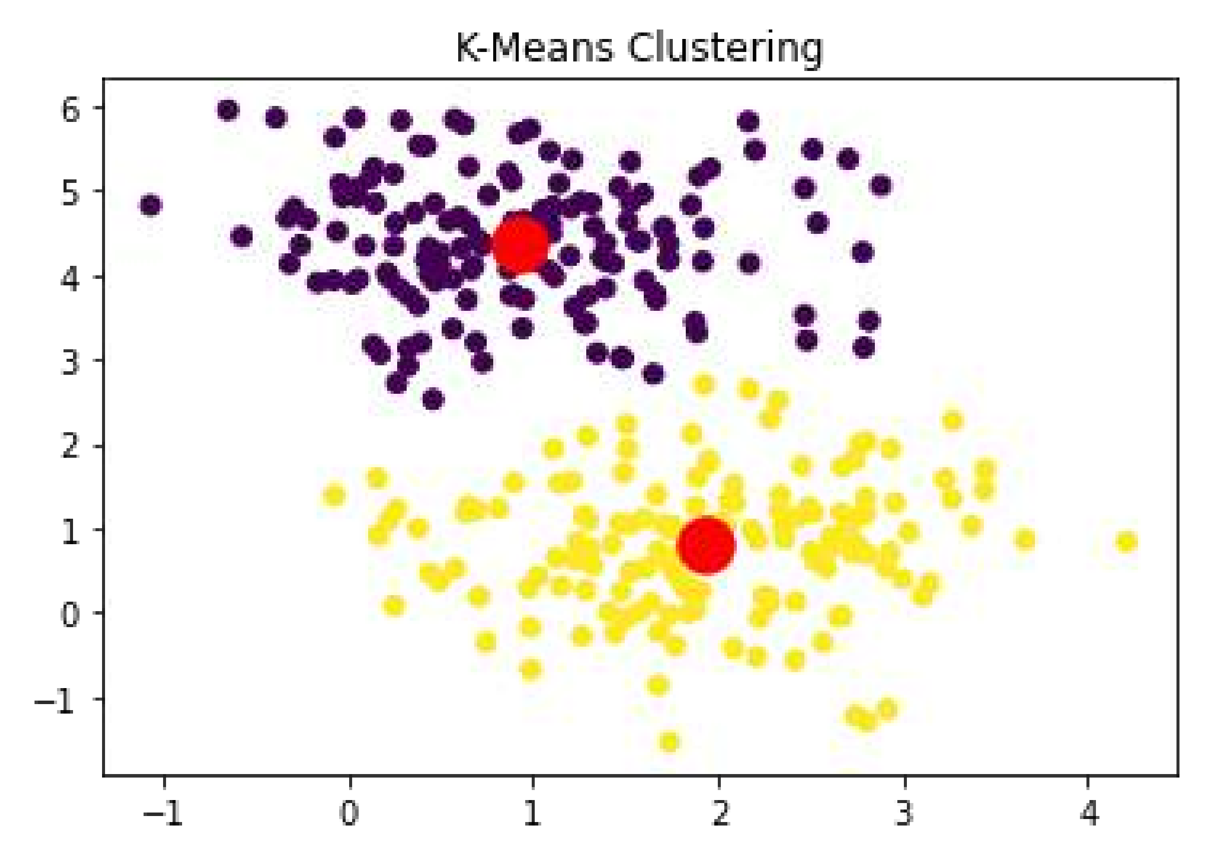



Y nuevamente movemos los centroides al centro de los nuevos datos asignados a cada uno de ellos:



Este proceso continúa hasta que no haya cambios en la asignación.

Al final, tenemos *K* grupos, cada uno representado por su centroide. Cada punto de datos pertenece al grupo con el centroide más cercano.

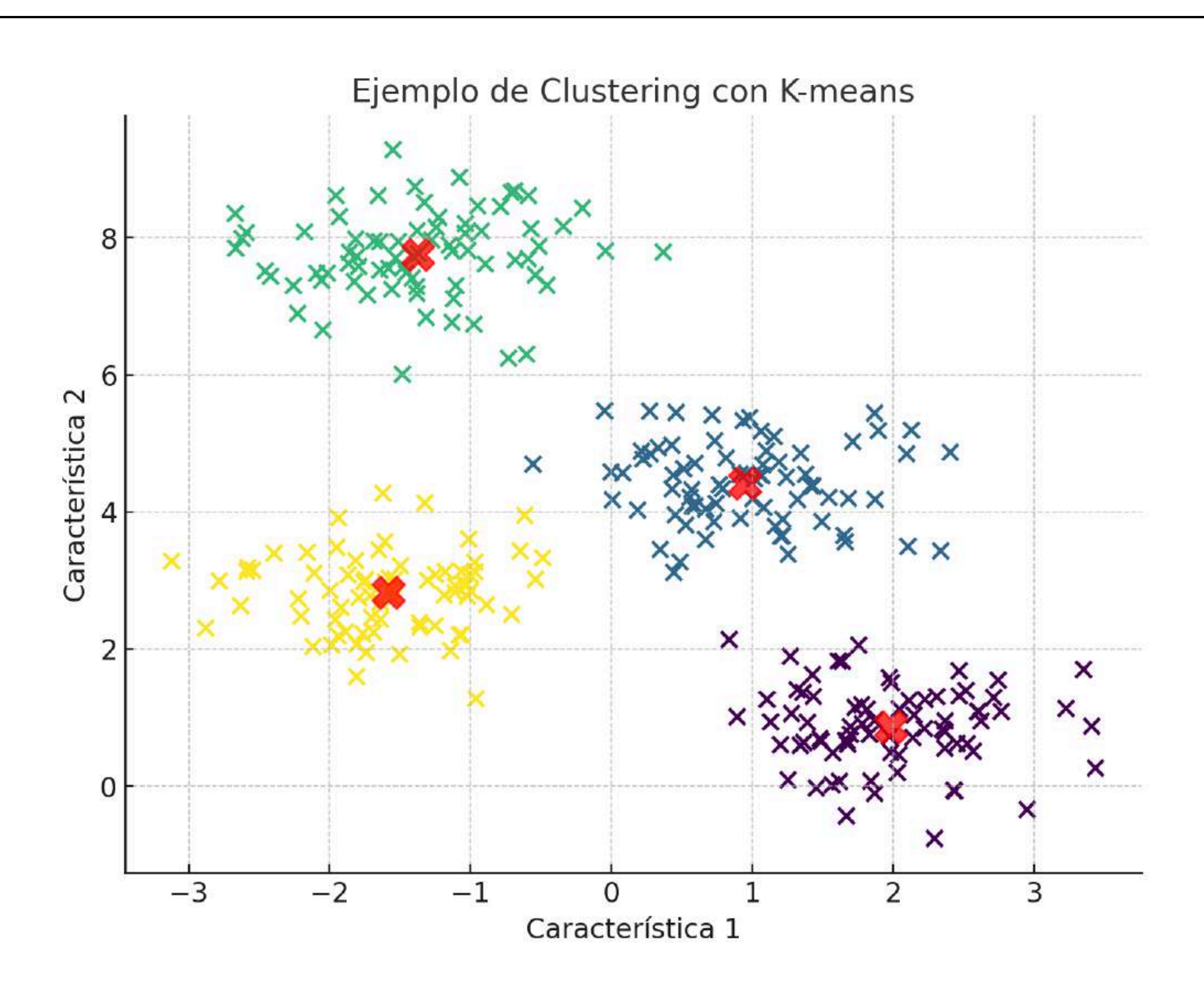


Este ejemplo puntual es con dos centroides (K=2) porque nuestra intuición viendo la nube de puntos inicial era que posiblemente existían dos conjuntos de datos distinguibles, pero en la práctica *K-Means* puede encontrar muchos más clusters.

Existen métodos[49] para automatizar la decisión de qué número asignar a *K*, pero excede el alcance de este libro. Con el *Tercer Paso* seguramente vas a llegar a ese conocimiento en muy poco tiempo.

Acá te dejo un gráfico con un un ejemplo visual de otro conjunto de datos al que le hemos aplicado un algoritmo K-Means con K=4:

[49] Algunos métodos populares para determinar el número óptimo de clusters incluyen el método del codo (elbow method), que analiza cómo cambia la varianza explicada al aumentar K; el coeficiente de silueta (silhouette score), que mide qué tan similar es cada punto a su propio cluster en comparación con otros clusters; y criterios estadísticos como BIC (Bayesian Information Criterion) o AIC (Akaike Information Criterion). También existen técnicas de validación cruzada especialmente adaptadas para clustering.



Esta imagen muestra un ejemplo de clustering utilizando el algoritmo K-means. Se observan cuatro grupos distintos de puntos de datos, cada uno representado por un color diferente. Cada grupo representa un cluster identificado por el algoritmo. Los puntos están distribuidos en un espacio bidimensional definido por dos características. Los puntos rojos más grandes en el centro de cada cluster representan los centroides, que son los puntos medios alrededor de los cuales se agrupan los datos. La clara separación entre los clusters demuestra la eficacia del algoritmo K-means en este caso para identificar patrones naturales en los datos y agruparlos en categorías distintas basadas en su similitud en el espacio de características.

En la práctica, K-means es ampliamente utilizado por su simplicidad y eficiencia en diversos contextos empresariales y científicos. Es especialmente popular en segmentación de clientes, donde las empresas agrupan a sus usuarios según patrones de comportamiento (como hábitos de compra, interacciones con el sitio web, o respuesta a campañas de marketing) para personalizar estrategias comerciales. También se utiliza frecuentemente en análisis de imágenes para reducir el número de colores, en sistemas de recomendación para agrupar usuarios con preferencias similares, y en control de calidad para detectar productos defectuosos según sus características. Sin embargo, es crucial recordar sus limitaciones: asume que los clusters son circulares y de tamaño similar, y requiere especificar el número de grupos de antemano. Por eso, en aplicaciones del mundo real, suele usarse como punto de partida para el análisis, frecuentemente complementado con otros métodos de clustering cuando se necesita mayor precisión o cuando los datos tienen estructuras más complejas.

- **Algoritmo de agrupamiento jerárquico**: el *hierarchical clustering* es una técnica de análisis de datos

que crea una estructura jerárquica de *clusters*.

Imaginá que queremos agrupar diferentes animales según sus características. Empezamos considerando cada animal como su propio grupo: un gato, un tigre, un lobo y un perro.



En el primer paso, el algoritmo podría juntar al gato y al tigre en un grupo porque son los más similares (ambos son felinos). Luego, uniría al perro y al lobo en otro grupo (ambos son cánidos). Finalmente, estos dos grupos se unirían en un grupo más grande de "carnívoros". Este proceso de agrupamiento paso a paso, donde vamos uniendo los elementos o grupos más similares, crea una estructura jerárquica que se puede visualizar como un árbol o "dendrograma", mostrándonos las relaciones y niveles de similitud entre todos los elementos.

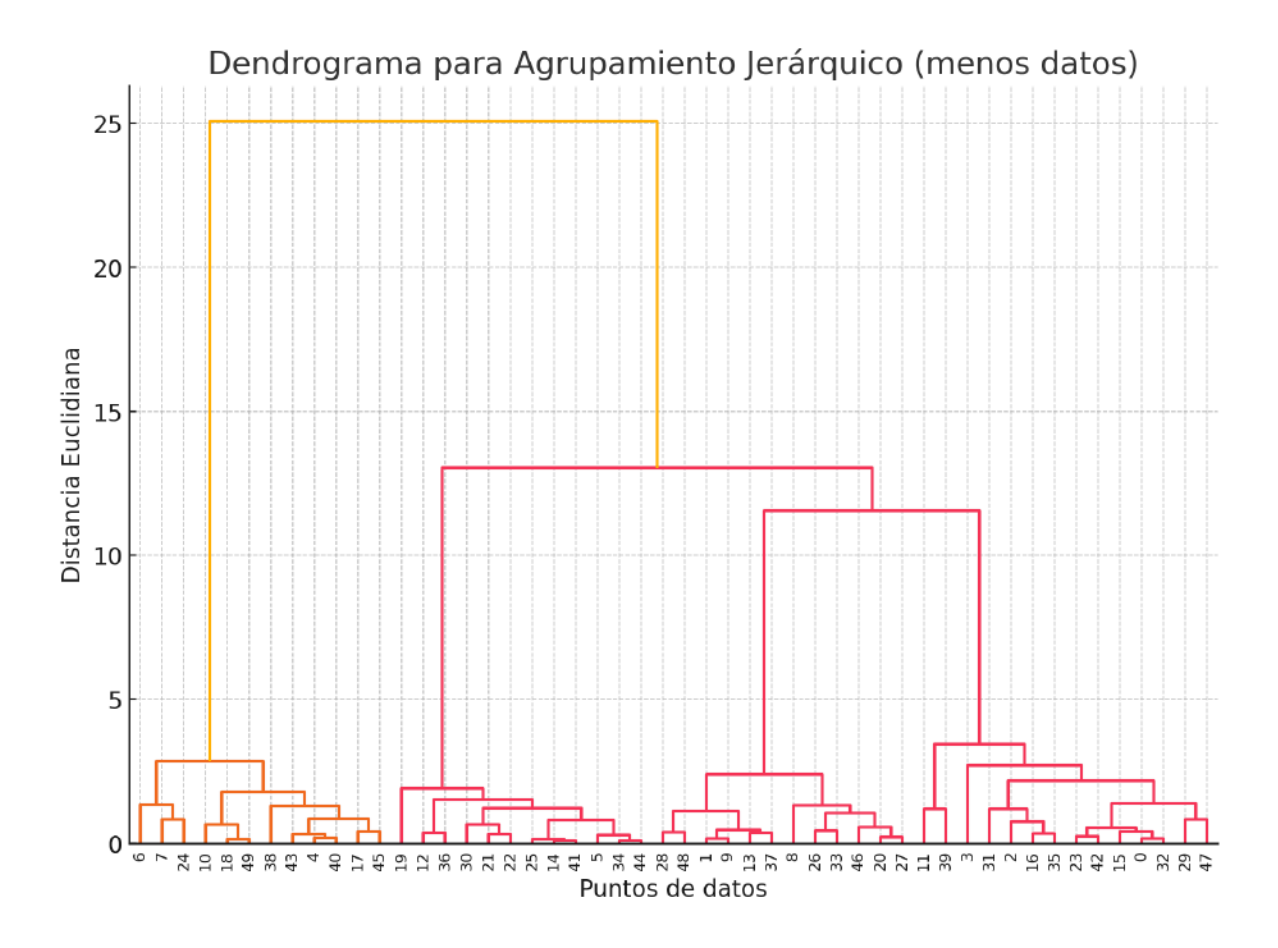
El algoritmo empieza tratando cada punto de datos como un *cluster* individual y luego, de manera iterativa, combina los clusters más similares hasta que todos los puntos pertenecen a un único *cluster* o se alcanza un número predefinido de *clusters*. Este proceso puede visualizarse como un árbol o dendrograma, donde las ramas representan las fusiones de *clusters* (ver página siguiente). Existen dos enfoques principales: *aglomerativo* (de abajo hacia arriba) y *divisivo* (de arriba hacia abajo). El método es valioso por su capacidad para revelar la estructura inherente de los datos sin requerir un número predefinido de *clusters* (como en K-Means), lo que lo hace útil en diversos campos como biología, marketing y análisis social. Sin embargo, puede ser computacionalmente costoso para grandes conjuntos de datos.

Podemos entenderlo mejor con otra analogía de cómo organizaríamos libros en una biblioteca:

1. **Enfoque Aglomerativo** (de abajo hacia arriba):
   - Comenzamos con cada libro como su propia categoría
   - Primero juntamos los libros más similares en pequeños grupos (por ejemplo, todos los libros de *El Señor de los Anillos*)
   - Luego unimos estos grupos en categorías más grandes (todos los libros de fantasía)
   - Finalmente, formamos categorías aún más grandes (toda la literatura juvenil)

2. **Enfoque Divisivo** (de arriba hacia abajo):
    - Comenzamos con todos los libros en un solo grupo
    - Dividimos primero en ficción y no ficción
    - Luego subdividimos ficción en fantasía, romance, misterio, etc.
    - Y así sucesivamente hasta llegar al nivel de detalle deseado

La ventaja principal es que podemos ver la relación entre grupos a diferentes niveles, como en un árbol genealógico, y no necesitamos decidir de antemano cuántos grupos queremos. Sin embargo, al igual que organizar una biblioteca muy grande, puede llevar mucho tiempo cuando hay muchos datos.



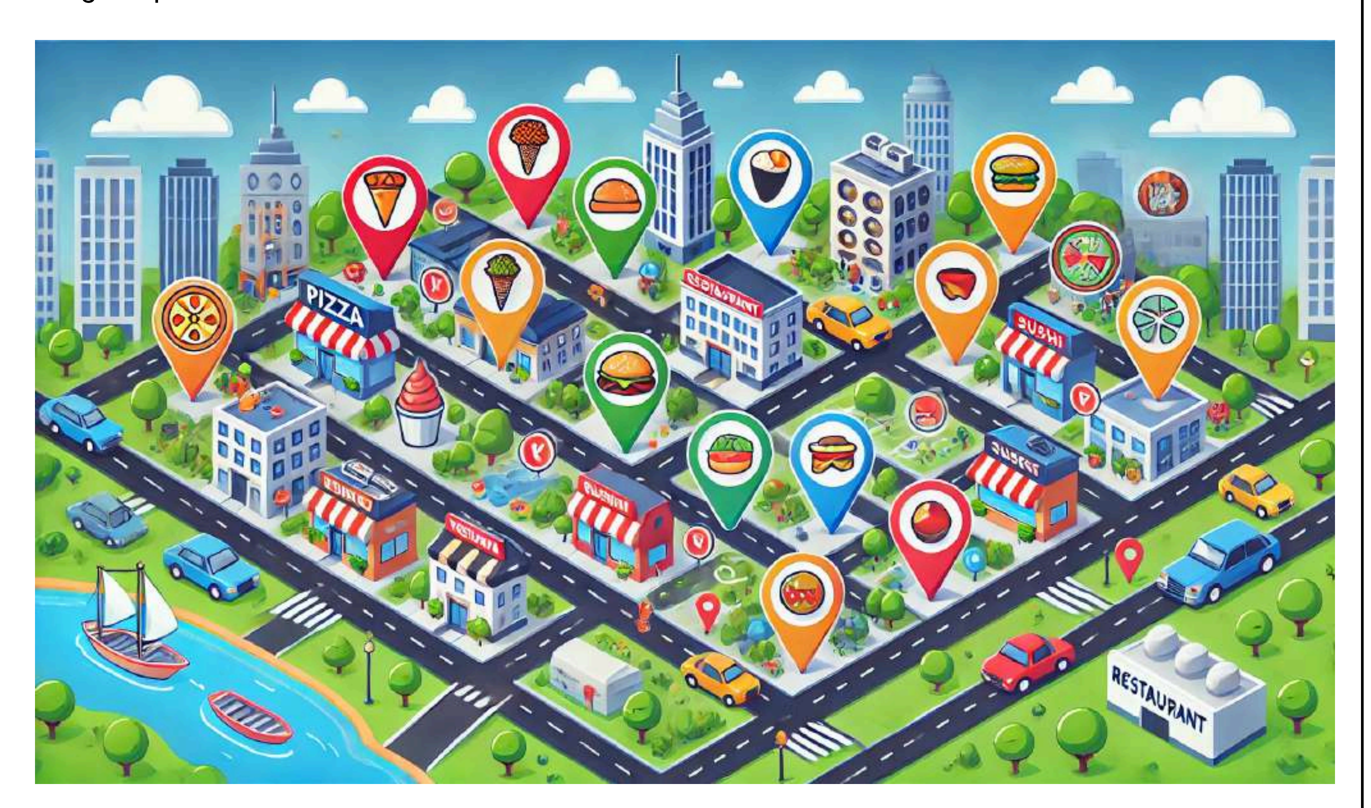
Esta imagen muestra un dendrograma, que es una representación visual del proceso de agrupamiento jerárquico. Las líneas verticales en la parte inferior representan puntos de datos individuales, mientras que las líneas horizontales indican las fusiones entre clusters a diferentes niveles de similitud o distancia. La altura de las conexiones horizontales representa la distancia o disimilitud entre los clusters que se están fusionando. Se pueden observar claramente dos grupos principales (uno en naranja y otro en rosado) que se unen en la parte superior del gráfico a una distancia euclidiana de aproximadamente 25. Dentro de estos grupos principales, especialmente en el grupo rosado, se pueden ver subgrupos más pequeños que se forman a distancias menores. Este dendrograma permite visualizar la estructura jerárquica de los datos y determinar el número óptimo de clusters según el nivel de corte elegido en la jerarquía.

En la práctica, el clustering jerárquico es especialmente valioso cuando necesitamos entender la estructura multinivel de nuestros datos o cuando no conocemos a priori el número de grupos que queremos formar. Es ampliamente utilizado en biología para clasificar especies y construir árboles filogenéticos, en análisis de documentos para organizar textos por temas y subtemas, y en estudios de mercado para segmentar clientes en grupos y subgrupos cada vez más específicos. Una de sus grandes ventajas es que produce el dendrograma, esta visualización intuitiva que permite a los stakeholders "cortar" el árbol a diferentes niveles según sus necesidades: pueden elegir tener pocos grupos grandes o muchos grupos pequeños sin necesidad de reentrenar el modelo. Sin embargo, su

costo computacional es mayor que el de K-means, por lo que en conjuntos de datos muy grandes suele aplicarse sobre una muestra representativa o se combina con otras técnicas de clustering más eficientes.

- **DBSCAN (Density-Based Spatial Clustering of Applications with Noise)**: es un algoritmo de agrupamiento basado en la *densidad* de los datos. Este método agrupa puntos que están estrechamente *empaquetados* en el espacio, marcando como outliers los puntos en regiones de baja densidad.

  Imaginá que estamos analizando la ubicación de restaurantes en una ciudad

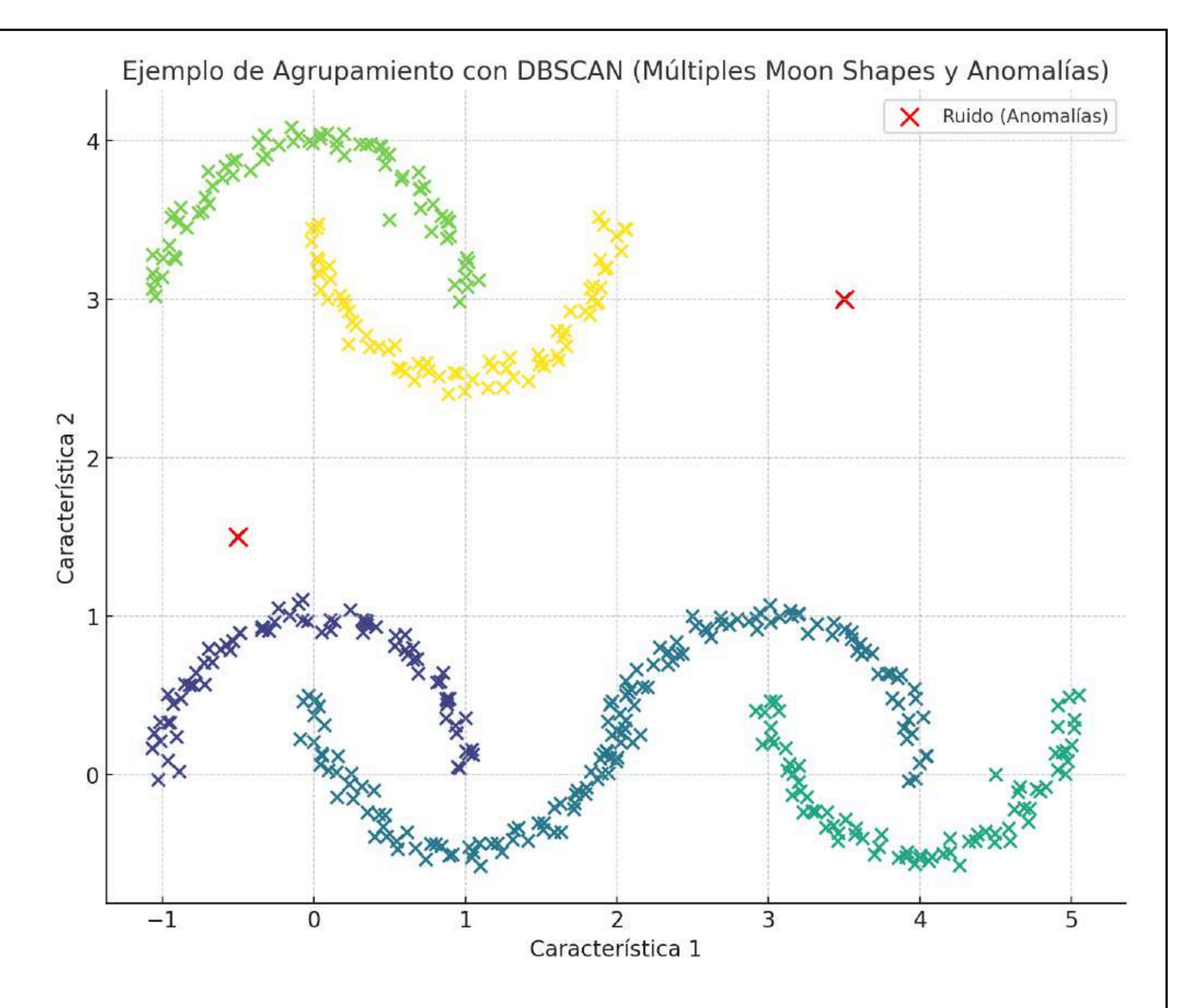


DBSCAN identificaría naturalmente los barrios gastronómicos (zonas con alta densidad de restaurantes cercanos entre sí, como el microcentro o el barrio chino) como clusters distintos, mientras que identificaría a restaurantes aislados (como un restaurante solitario en una zona residencial) como outliers. El algoritmo no necesita que le especifiquemos cuántos barrios gastronómicos buscar - encontrará automáticamente todos los grupos donde los restaurantes están lo suficientemente cerca unos de otros, independientemente de la forma que tengan estas zonas en el mapa e ignorando los restaurantes que están demasiado aislados.

DBSCAN requiere dos parámetros principales:

- la distancia máxima entre dos puntos para ser considerados vecinos (epsilon) y
- el número mínimo de puntos para formar un cluster denso (minPts).

El algoritmo comienza con un punto arbitrario y expande el cluster a todos los puntos densamente conectados. A diferencia de K-means, DBSCAN no necesita que se especifique el número de *clusters* de antemano y puede descubrir clusters de formas arbitrarias. Es particularmente efectivo para identificar clusters de formas no convencionales y para detectar anomalías en los datos, lo que lo hace útil en aplicaciones como la detección de fraude o el análisis de patrones espaciales.



Esta imagen ilustra el resultado de aplicar el algoritmo DBSCAN a un conjunto de datos bidimensional. Se observan cinco clusters principales con formas de media luna o "moon shapes", cada uno representado por un color diferente (verde, amarillo, azul oscuro, azul claro y verde-azulado). Estas formas no convencionales demuestran la capacidad de DBSCAN para identificar clusters de formas arbitrarias, algo que algoritmos como K-means no podrían hacer eficazmente. Además, se pueden ver dos puntos rojos aislados, que el algoritmo ha clasificado como ruido o anomalías. Estos puntos están en áreas de baja densidad, lejos de los clusters principales. La imagen resalta la efectividad de DBSCAN para detectar patrones basados en la densidad de los datos y su habilidad para identificar outliers, lo que lo hace útil en escenarios donde los clusters no tienen una forma esférica y donde la detección de anomalías es importante.

Podemos entenderlo mejor con esta analogía de grupos de personas en una plaza:

- **Cómo funciona**:
    - El algoritmo necesita dos reglas simples:
        1. Qué tan cerca deben estar las personas para considerarlas parte del mismo grupo
        2. Cuántas personas mínimo se necesitan para formar un grupo
- **El algoritmo clasifica los puntos en tres tipos**:
    - **Puntos núcleo**: Como las personas en el centro de un grupo (tienen muchos vecinos cercanos)
    - **Puntos borde**: Como las personas en el borde del grupo (tienen pocos vecinos pero están cerca de un grupo)

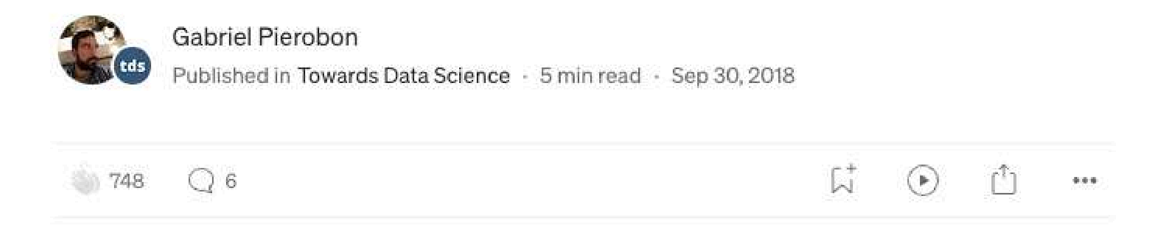
- **Ruido**: Como personas solitarias alejadas de cualquier grupo

La ventaja principal de DBSCAN es que puede encontrar grupos de cualquier forma (como personas formando un círculo o una fila), y no necesita que le digamos cuántos grupos buscar. También es muy bueno identificando "casos atípicos" (como una persona parada sola lejos de las multitudes).

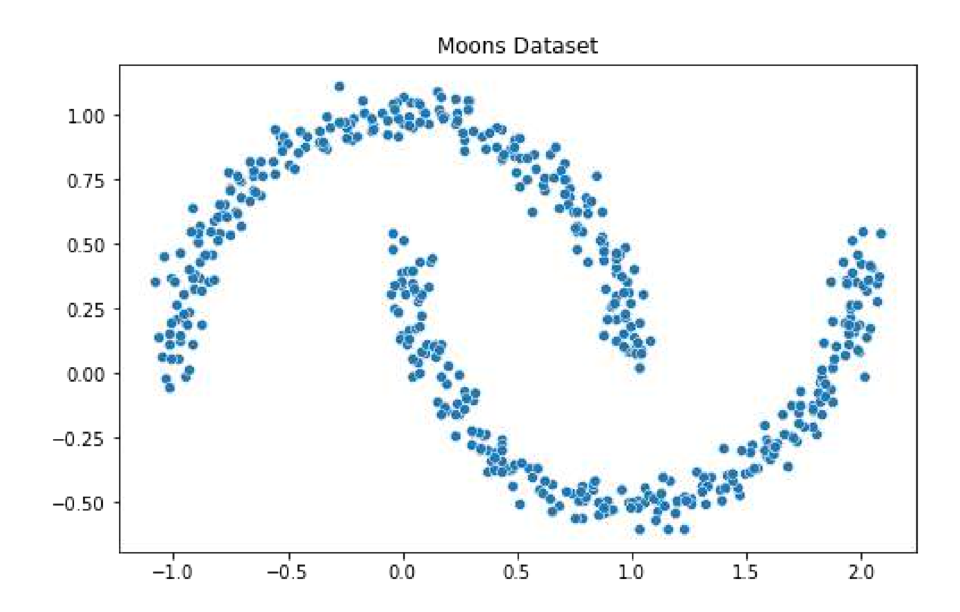
Uno de los primeros artículos que publiqué en Medium fue sobre DBSCAN[50]:



DBSCAN clustering for data shapes k-means can't handle well (in Python)

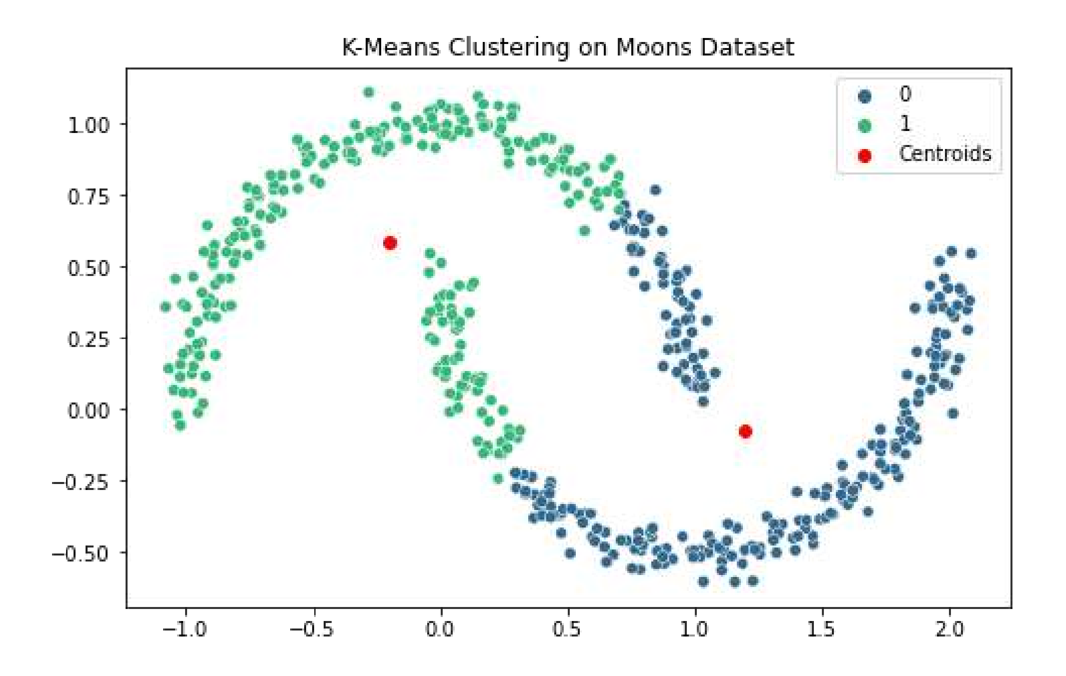


En este artículo hago una demostración de la ventaja de utilizar DBSCAN por sobre K-Means para clusters que tienen formas muy específicas, como por ejemplo el que ya vimos.



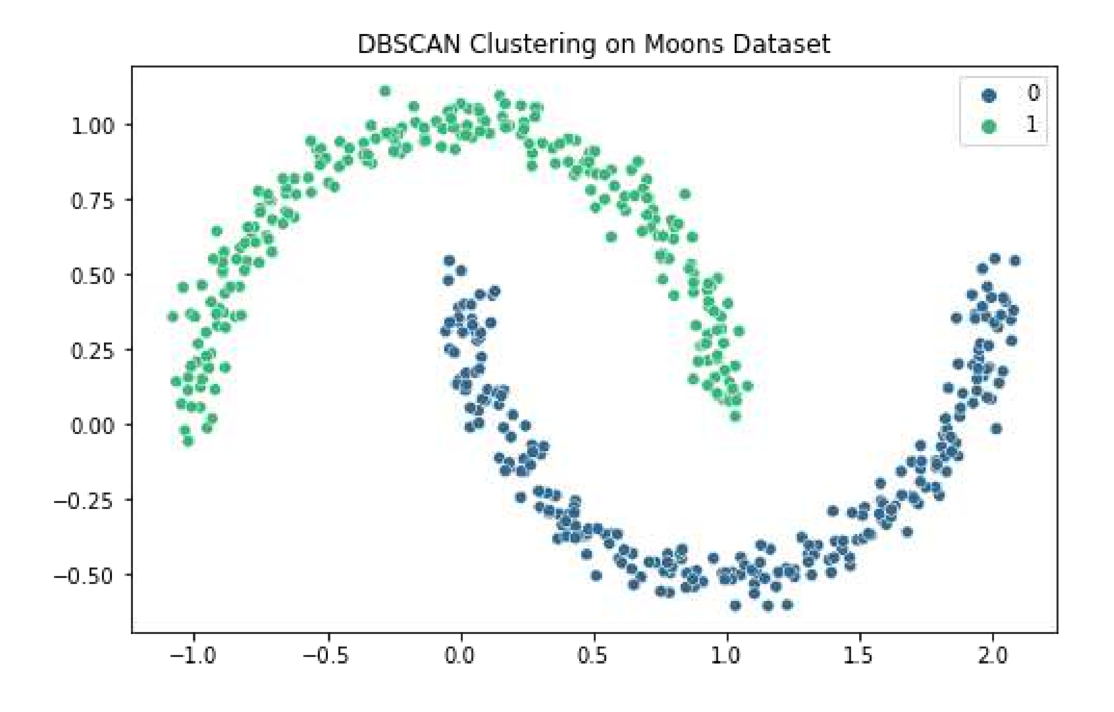
Donde K-Means con K=2 generaría este resultado:

50 https://medium.com/towards-data-science/dbscan-clustering-for-data-shapes-k-means-cant-handle-well-in-python-6be89af4e6ea



Claramente podemos ver que este algoritmo por su funcionamiento no es capaz de asignar los datos a los clusters correctos, sino que lo hace por proximidad a los *centroides* siguiendo el proceso de convergencia de K-Means que vimos anteriormente.

Por otro lado, con DBSCAN logramos el *clustering* perfecto:



En la práctica, DBSCAN es particularmente valioso cuando trabajamos con datos que contienen ruido y clusters de formas irregulares, situaciones donde K-means podría fallar. Es extensamente utilizado en sistemas de geolocalización para identificar zonas de interés (como áreas comerciales o puntos turísticos), en análisis de redes sociales para detectar comunidades, y en sistemas de detección de

anomalías en datos de sensores o transacciones financieras. Su capacidad para identificar automáticamente outliers lo hace especialmente útil en tareas de limpieza de datos y detección de fraude. Sin embargo, su rendimiento depende críticamente de la elección de dos parámetros: la distancia máxima entre puntos (epsilon) y el número mínimo de puntos para formar un cluster (minPts), que generalmente requieren conocimiento del dominio o experimentación para ser ajustados correctamente.

- **PCA (Principal Component Analysis,** Análisis de Componentes Principales**)**: es una técnica estadística fundamental para la reducción de dimensionalidad en el análisis de datos. Este método transforma un conjunto de variables posiblemente correlacionadas en un conjunto menor de variables no correlacionadas llamadas *componentes principales*. La explicación sencilla de PCA es que es una técnica que nos permite reducir el tamaño de nuestros datos con la menor pérdida de información posible (como un *zip* de nuestros datos) y es muy útil cuando tenemos datasets excesivamente grandes. Técnicamente el proceso implica encontrar las direcciones de máxima varianza en los datos de alta dimensión y proyectar los datos sobre estas direcciones. Los *componentes principales* se ordenan por la cantidad de varianza que explican, permitiendo reducir la dimensionalidad al retener sólo los componentes más significativos. PCA es ampliamente utilizado en reconocimiento de patrones, compresión de datos, y visualización de datos multidimensionales. Es particularmente útil para simplificar conjuntos de datos complejos, eliminar ruido y redundancia, y revelar estructuras subyacentes en los datos, facilitando así su interpretación y análisis posterior.

  Con PCA no buscamos *clusters* en los datos, sino transformarlos a una versión más valiosa y manejable de ellos mismos.

  Podemos entenderlo mejor con algunas analogías:

  - **Como una foto con compresión**:
    - Imaginá que tenés una foto de alta resolución que ocupa mucho espacio
    - PCA es como crear una versión comprimida que mantiene los detalles más importantes
    - Descarta la información menos relevante pero preserva la esencia de la imagen
  - **Como resumir un libro**:
    - El libro original tiene por ejemplo 100 dimensiones (variables)
    - PCA encuentra las "ideas principales" (componentes principales)
    - Podríamos quedarnos con las 10 ideas más importantes que capturan la mayoría del contenido

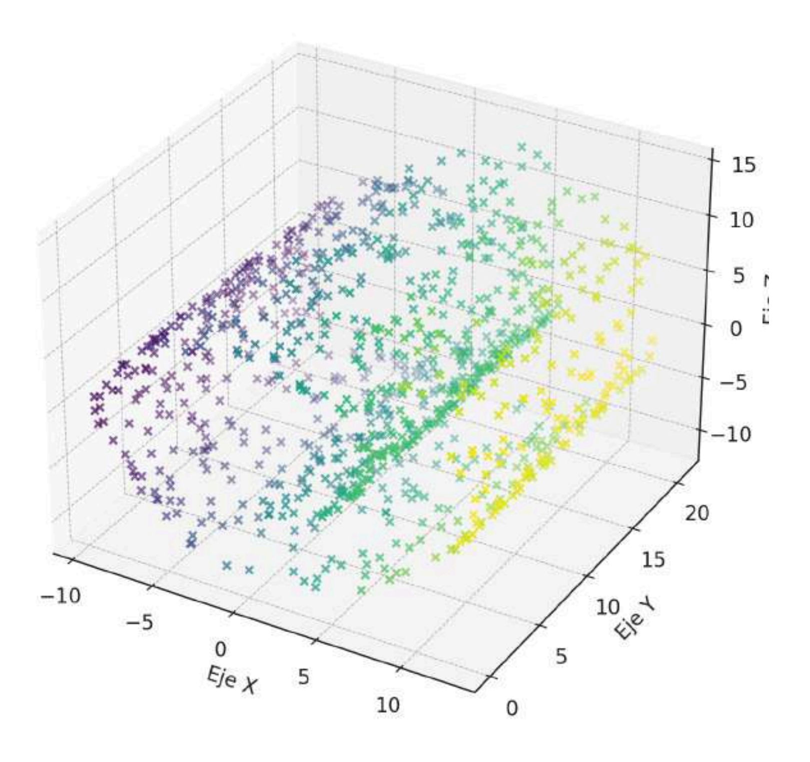
  La clave está en que PCA:

  - Identifica los patrones más importantes en los datos
  - Elimina la información redundante (como cuando varias variables dicen casi lo mismo)
  - Nos permite trabajar con una versión simplificada pero informativa de nuestros datos
  - Facilita la visualización de datos complejos al reducirlos a menos dimensiones (un ejemplo de esto en las siguientes páginas)

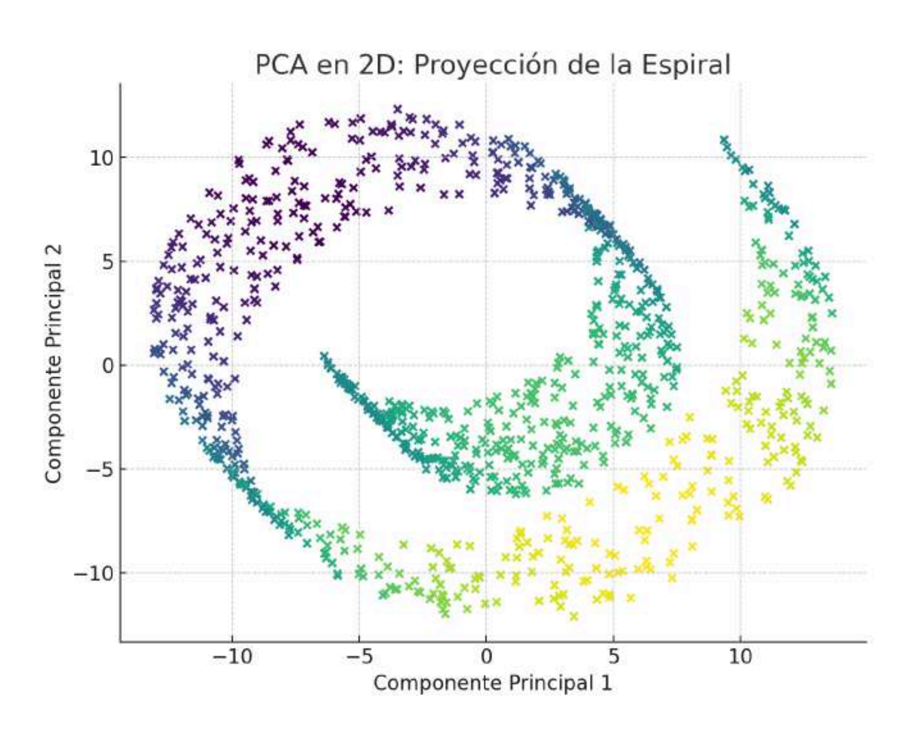
  Es como tener un "resumen inteligente" que mantiene la información más relevante mientras reduce significativamente el tamaño de los datos.

  Las siguientes imágenes ilustran el proceso y resultado de aplicar PCA a un conjunto de datos tridimensional. El primer gráfico muestra los datos originales en 3D, que forman una estructura en espiral. Cada punto está representado por un color que indica su posición en la espiral.





El segundo gráfico muestra el resultado de aplicar PCA para reducir estos datos de 3D a 2D. La proyección 2D mantiene la estructura espiral de los datos originales, demostrando cómo PCA puede preservar las características principales de los datos incluso al reducir su dimensionalidad. Los colores de los puntos se mantienen consistentes entre ambas visualizaciones, permitiendo observar cómo se ha "desenrollado" la espiral 3D en un plano 2D. Esta transformación facilita el análisis y visualización de los datos, manteniendo la mayor parte de la información relevante en menos dimensiones.



Aplicaciones del Aprendizaje No Supervisado



Segmentación de Clientes

Divide la base de clientes en grupos homogéneos para personalizar estrategias de marketing.



Agrupación de Noticias

Organiza artículos en grupos temáticos para facilitar el acceso a la información.



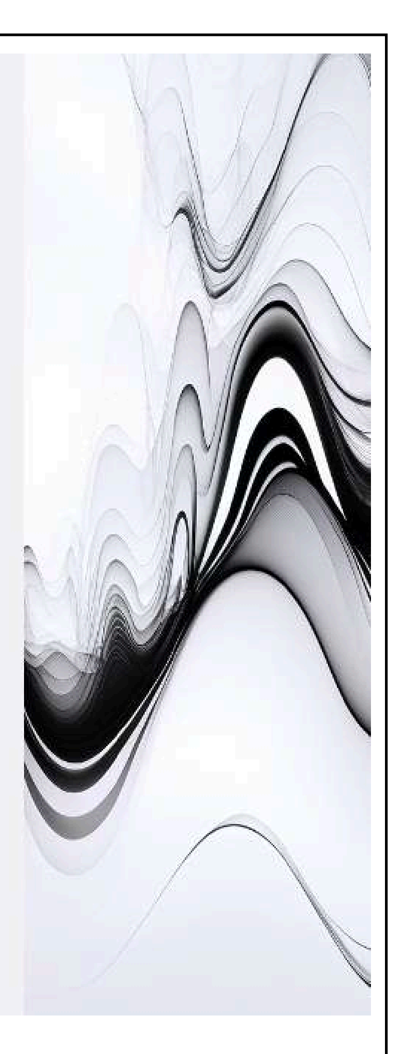
Detección de Anomalías

Identifica comportamientos anómalos en redes que podrían indicar ataques cibernéticos.



Bioinformática

Identifica estructuras en secuencias de ADN, ayudando en el descubrimiento de fármacos.



Ejemplo de aprendizaje no supervisado: Segmentación de clientes utilizando K-Means

En este ejemplo, vamos a simular un caso real en el que una tienda online desea segmentar su base de datos de clientes para identificar grupos con patrones de comportamiento similares. Esta segmentación permitirá diseñar campañas de marketing más específicas y relevantes, algo que es fundamental en el actual entorno competitivo donde los consumidores esperan experiencias cada vez más personalizadas. La capacidad de dividir la base de clientes en grupos significativos le permite a las empresas optimizar sus recursos de marketing, mejorar las tasas de conversión y aumentar significativamente el retorno de inversión en sus campañas promocionales.

Esta estrategia de segmentación no solo beneficia a las empresas en términos de eficiencia y rentabilidad, sino que también mejora sustancialmente la experiencia del cliente. Al recibir comunicaciones y ofertas más relevantes para sus necesidades específicas, los clientes desarrollan una mayor lealtad hacia la marca, lo que se traduce en un incremento del valor del ciclo de vida del cliente (*Customer Lifetime Value*) y en relaciones comerciales más duraderas. Además, la segmentación proporciona *insights* valiosos sobre el comportamiento del consumidor, permitiendo a las empresas anticiparse a las tendencias del mercado y adaptar sus estrategias de manera más efectiva.

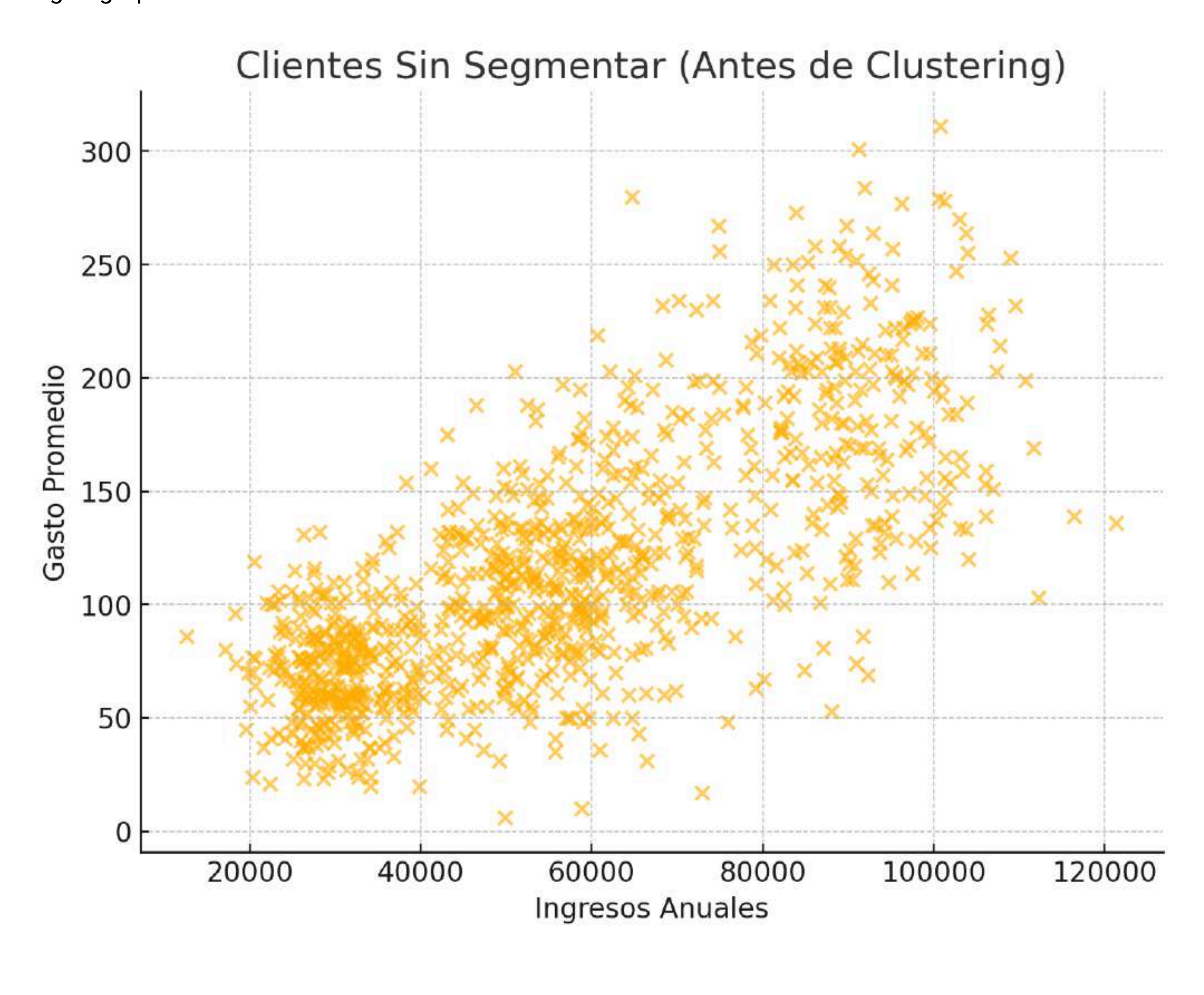
Paso 1: Conjunto de datos inicial

Partimos de una base de datos que contiene información demográfica y de comportamiento de compra de los clientes. Cada registro incluye atributos como la edad, ingresos anuales, frecuencia de compra, el gasto promedio en cada transacción y el tiempo promedio que los clientes pasan navegando en el sitio web.

Muestra de datos iniciales (primeras 5 filas):

Edad	Ingresos Anuales	Frecuencia de Compra	Gasto Promedio	Tiempo en Sitio
24	45,000	14	80	30
36	62,000	10	150	20
59	50,000	5	100	28
27	72,000	12	120	25
41	80,000	9	180	15

Estos datos iniciales aún no contienen información sobre la relación entre clientes, es decir, no sabemos si hay patrones ocultos que los agrupan en segmentos naturales. Si observamos el gráfico siguiente, donde se muestran los ingresos anuales y el gasto promedio, parece que hay cierta dispersión, pero no podemos distinguir grupos claros:



Paso 2: *Clustering* utilizando K-Means

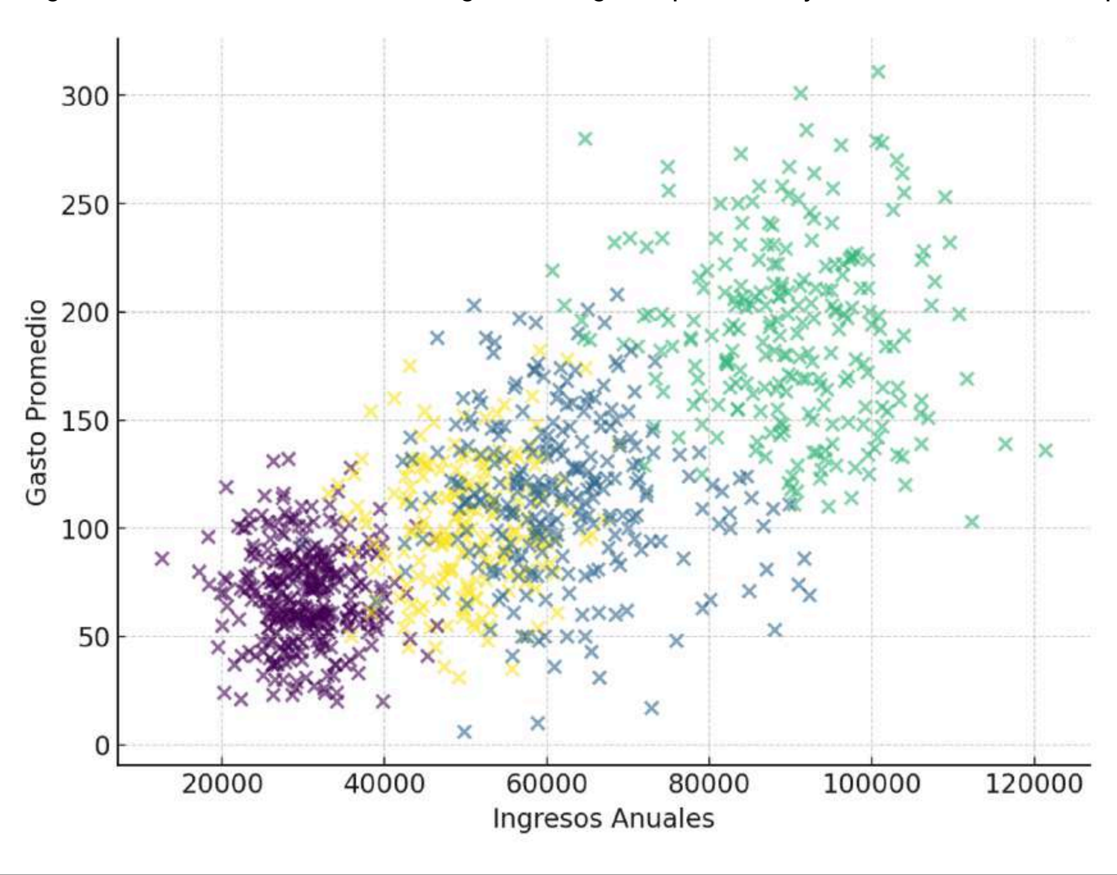
Para descubrir patrones ocultos y segmentar automáticamente a los clientes en grupos, utilizaremos el algoritmo de *K-Means*. Como ya vimos, este algoritmo es ideal cuando se quiere dividir una población en varios *clusters* de acuerdo con su similitud en múltiples variables. Recordá que K-Means funciona agrupando los datos en *K* clusters distintos según la distancia entre los puntos, es decir, aquellos datos más cercanos entre sí en el espacio de características.

K-Means es particularmente útil en este caso porque:

- Es simple y eficiente para dividir grandes conjuntos de datos.
- Es flexible en cuanto al número de clusters que se quiere crear.
- Ayuda a descubrir patrones que no son visibles a simple vista.

El primer paso para aplicar K-Means es decidir cuántos clusters (K) queremos[51]. En este ejercicio hemos elegido K=4, asumiendo con cierto conocimiento del negocio que los clientes podrían dividirse en cuatro grupos diferentes de comportamiento .

El resultado del *clustering*, mostrado a continuación, revela grupos más definidos[52]. El algoritmo ha separado los clientes según características como sus ingresos, el gasto promedio y su frecuencia de compra.



[51] Recordá que en páginas anteriores, en una nota al pie, se mencionó que hay métodos para encontrar el *K óptimo*.

[52] Considerá que en este gráfico se muestran solo dos dimensiones: Gasto Promedio e Ingresos Anuales, pero en nuestros datos tenemos más dimensiones como Frecuencia de Compra y Tiempo en Sitio. Esta es la razón por la cual en el gráfico hay puntos que pertenecen a diferentes *clusters* pero entrecruzados. Dicho de otro modo, aunque los puntos amarillos y los azules se mezclen un poco en este gráfico, la razon por la que pertenecen a diferentes clusters es que en Frecuencia de Compra y/o Tiempo en Sitio son suficientemente diferentes. La limitación es que acá no puedo poner un gráfico en 3D o 4D.

Paso 3: Interpretación de los *Clusters*

Después de aplicar K-Means, podríamos analizar los cuatro grupos obtenidos para encontrar una descripción adecuada a cada segmento. Cada grupo representaría un conjunto de clientes con patrones de comportamiento similares:

- **Cluster 1 (Violeta)**: *Este grupo incluiría a clientes jóvenes con ingresos bajos, que realizan compras frecuentes pero con un gasto promedio reducido. Serían clientes más sensibles a las ofertas y probablemente más interesados en productos económicos o promociones frecuentes.*
- **Cluster 2 (Amarillo)**: *Compradores con ingresos medios y un gasto promedio moderado, que realizan compras de forma regular. Estos podrían ser jóvenes profesionales con ingresos estables que buscan un balance entre calidad y precio.*
- **Cluster 3 (Azul):** *Clientes de edad media y alta, con ingresos elevados y un gasto promedio alto. Aunque compran con menos frecuencia, suelen hacer compras de mayor valor, lo que sugiere que podrían estar interesados en productos premium o de alta gama.*
- **Cluster 4 (Verde):** *Este grupo estaría compuesto por clientes de ingresos medios-altos, que compran regularmente y tienen un gasto promedio alto. Estos clientes serían valiosos para la tienda, ya que combinan frecuencia y valor en sus compras.*

Muestra de datos segmentados (primeras 5 filas):

Edad	Ingresos Anuales	Frecuencia de Compra	Gasto Promedio	Tiempo en Sitio	Cluster
24	30,000	18	75	40	1
36	62,000	12	140	20	2
58	50,000	4	80	30	3
27	72,000	10	110	28	4
41	90,000	6	180	15	4

La implementación del algoritmo *K-Means* nos permitió transformar una base de datos aparentemente homogénea en cuatro segmentos claramente diferenciados, cada uno con sus propias características y necesidades específicas. Esta segmentación no solo nos proporciona una mejor comprensión de nuestra base de clientes, sino que también nos permite desarrollar estrategias de marketing más efectivas y personalizadas.

Beneficios Inmediatos

- 🎯 **Marketing personalizado:** Capacidad para crear campañas específicas para cada segmento
- 🌟 **Mejor experiencia:** Comunicaciones más relevantes que aumentan la satisfacción del cliente
- 📊 **Optimización de recursos:** Asignación más eficiente del presupuesto de marketing
- 📈 **Mayor ROI:** Incremento en las tasas de conversión y el valor promedio de compra

Próximos pasos

1. Implementar campañas personalizadas por segmento
2. Monitorear el comportamiento y respuesta de cada grupo

3. Ajustar las estrategias según los resultados obtenidos

Esta segmentación representa el primer paso hacia una estrategia de marketing más inteligente y orientada a datos, que nos permitirá construir relaciones más sólidas con nuestros clientes mientras optimizamos nuestros recursos de marketing.

Con esto llegamos al final de nuestra sección sobre aprendizaje automático no supervisado y también concluimos la *mini* sección dedicada al *machine learning* en general (en conjunto con la anterior de aprendizaje supervisado), aunque volveremos a ver todos estos temas nuevamente en varias oportunidades. Espero que hayas aprendido las características principales de ambos métodos de entrenamiento de modelos de inteligencia artificial y que tengas un mejor contexto de **cómo, por qué y para qué** se aplican. Volverte una persona experta en *machine learning* dependerá de cuánto tiempo dediques al *Tercer Paso*, pero aún no nos podemos anticipar a esto porque todavía nos quedan muchísimos conceptos básicos de la IA que quiero que aprendas, empezando por un tipo de inteligencia artificial que no aprende de *etiquetas* ni de *distancia* entre los datos, sino que aprende de la misma forma que lo hacemos los humanos, mediante prueba y error interactuando con el entorno. Veamos de qué se trata el aprendizaje por refuerzo.

05. 🕹️ Aprendizaje por Refuerzo (Reinforcement Learning)



Aprendizaje por Refuerzo: Optimización a través de la Interacción

El aprendizaje por refuerzo es una técnica donde los agentes aprenden a tomar decisiones optimizadas interactuando con su entorno. Es crucial para sistemas autónomos y robótica, adaptándose a condiciones cambiantes.

El *Reinforcement Learning* o *Aprendizaje por Refuerzo* es tal vez a los ojos la forma más clara y natural de entender la inteligencia artificial porque refleja en gran medida cómo nosotros los humanos (y posiblemente una gran proporción de los seres vivos) aprendemos naturalmente: a través de la experiencia directa, prueba y error, y la búsqueda de recompensas. Incluso mirando en tiempo real un sistema que aprende por refuerzo es fácil de intuir lo que está sucediendo (cuando completes esta sección vas a ver por qué).

Personalmente y como un gran fanático de los videojuegos, el aprendizaje por refuerzo siempre me llamó la atención ver cómo estos sistemas pueden aprender a dominar juegos extremadamente complejos, aunque debo admitir que desafortunadamente no tuve demasiadas oportunidades profesionales para aplicarlo activamente. No obstante, con los progresos recientes en LLMs y en la IA basada en agentes (conceptos que veremos más adelante en detalle) encontré puntos de contacto en mi día a día que me permitieron comenzar cada vez más a indagar en esta disciplina tan fascinante[53].

---

[53] Para esta sección dependeré en gran medida de lo que pueda asistir en generar Claude o GPT 4 pero seguramente será una de estas secciones que modificaré con más frecuencia personalmente.

En lugar de aprender de un conjunto fijo de datos etiquetados (como vimos en el aprendizaje supervisado) o encontrar patrones en datos no etiquetados (como vimos en el aprendizaje no supervisado), el **aprendizaje por refuerzo** se centra en agentes que toman decisiones secuenciales para maximizar una recompensa acumulada a lo largo del tiempo. Estos agentes exploran su entorno, deciden, se equivocan, pero cada vez aprenden más hasta finalmente conseguir su objetivo final.

Es parecido a cómo aprendemos muchas habilidades en la vida, a través de prueba y error, guiados por *recompensas*. Podemos entenderlo mejor con la analogía de cómo un nene aprende a andar en bicicleta:

- **El proceso de aprendizaje**:
  - El nene (agente) interactúa con la bicicleta y el entorno
  - Toma decisiones: girar el manubrio, pedalear más rápido o más lento
  - Recibe "recompensas" naturales:
    - Positivas: mantener el equilibrio, llegar a su destino
    - Negativas: caerse, chocar
- **Características clave**:
  - No hay necesariamente un "instructor" que le diga exactamente qué hacer en cada momento
  - Aprende de la experiencia directa y sus consecuencias
  - Tiene que balancear entre:
    - *Explorar* nuevas acciones (probar cosas nuevas)
    - *Explotar* lo que ya sabe que funciona (usar técnicas probadas)

Con el tiempo, igual que el nene mejora sus habilidades en bicicleta, un *agente* aprende qué acciones llevan a mejores resultados en diferentes situaciones, optimizando su comportamiento para maximizar las recompensas a largo plazo.



Aplicaciones del Aprendizaje por Refuerzo

Juegos

Dominio de juegos de tablero y videojuegos complejos.

Robótica

Aprendizaje de tareas complejas como caminar y manipular objetos.

Sistemas de Recomendación

Adaptación dinámica a preferencias de usuarios para recomendaciones personalizadas.

Conceptos Clave

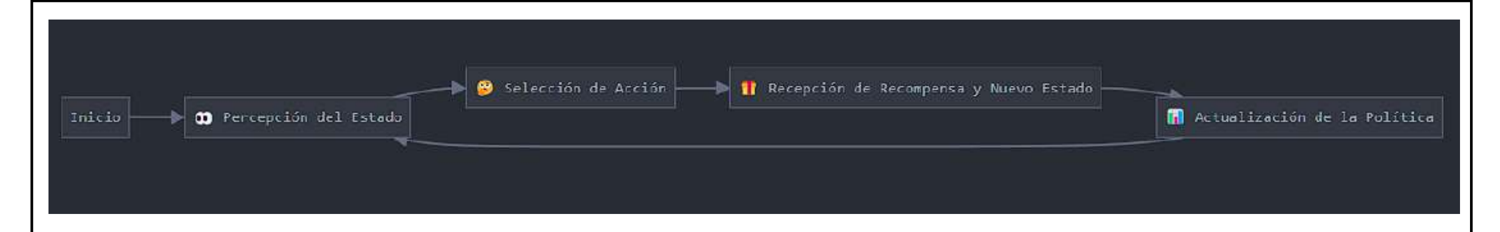
- 🤖 **Agente**: El *ente* que toma decisiones y aprende dentro del *entorno*.
- 🌍 **Entorno**: Todo lo que rodea al agente y con lo que interactúa.
- 📸 **Estado**: Una representación de la situación actual del *entorno* en un momento dado.
- 🎬 **Acción**: Las posibles *decisiones* o *movimientos* que el agente puede tomar.
- 🏆 **Recompensa**: La retroalimentación que el agente recibe del *entorno* después de tomar una acción. Puede ser positiva o negativa y guía el aprendizaje del agente.
- 📊 **Política**: Una estrategia que define qué acción tomar en cada estado.
- 📈 **Función de Valor**: Una función que estima el valor esperado de estar en un estado dado, considerando las recompensas futuras.
- 🎯 **Q-Valor (Valor de Acción)**: Una función que estima el valor esperado de tomar una acción en un estado específico.

En el ejemplo de aprender a andar en bicicleta:

- 🤖 **Agente**: El nene que está aprendiendo a andar en bicicleta
- 🌍 **Entorno**: La calle, el pavimento, los obstáculos, la pendiente del terreno
- 📸 **Estado**: La situación actual: velocidad de la bicicleta, inclinación, posición del manubrio, si está balanceado o no
- 🎬 **Acción**: Las decisiones que puede tomar: pedalear, frenar, girar el manubrio, inclinarse
- 🏆 **Recompensa**: Las consecuencias inmediatas: mantenerse en equilibrio (+1), avanzar (+1), caerse (-1)
- 📊 **Política**: La estrategia aprendida: "si empiezo a inclinarme a la derecha, debo girar el manubrio levemente hacia la derecha"
- 📈 **Función de valor**: Qué tan "bueno" es estar en cierta situación: "ir a velocidad moderada en terreno plano es mejor que ir muy lento en una pendiente"
- 🎯 **Q-Valor**: El valor esperado de cada acción: "si estoy perdiendo el equilibrio, girar el manubrio suavemente es mejor que girarlo bruscamente"

Todos estos elementos trabajan juntos para que el agente (el nene) aprenda gradualmente a través de la experiencia, igual que en cualquier sistema de aprendizaje por refuerzo.

Proceso de Aprendizaje por Refuerzo

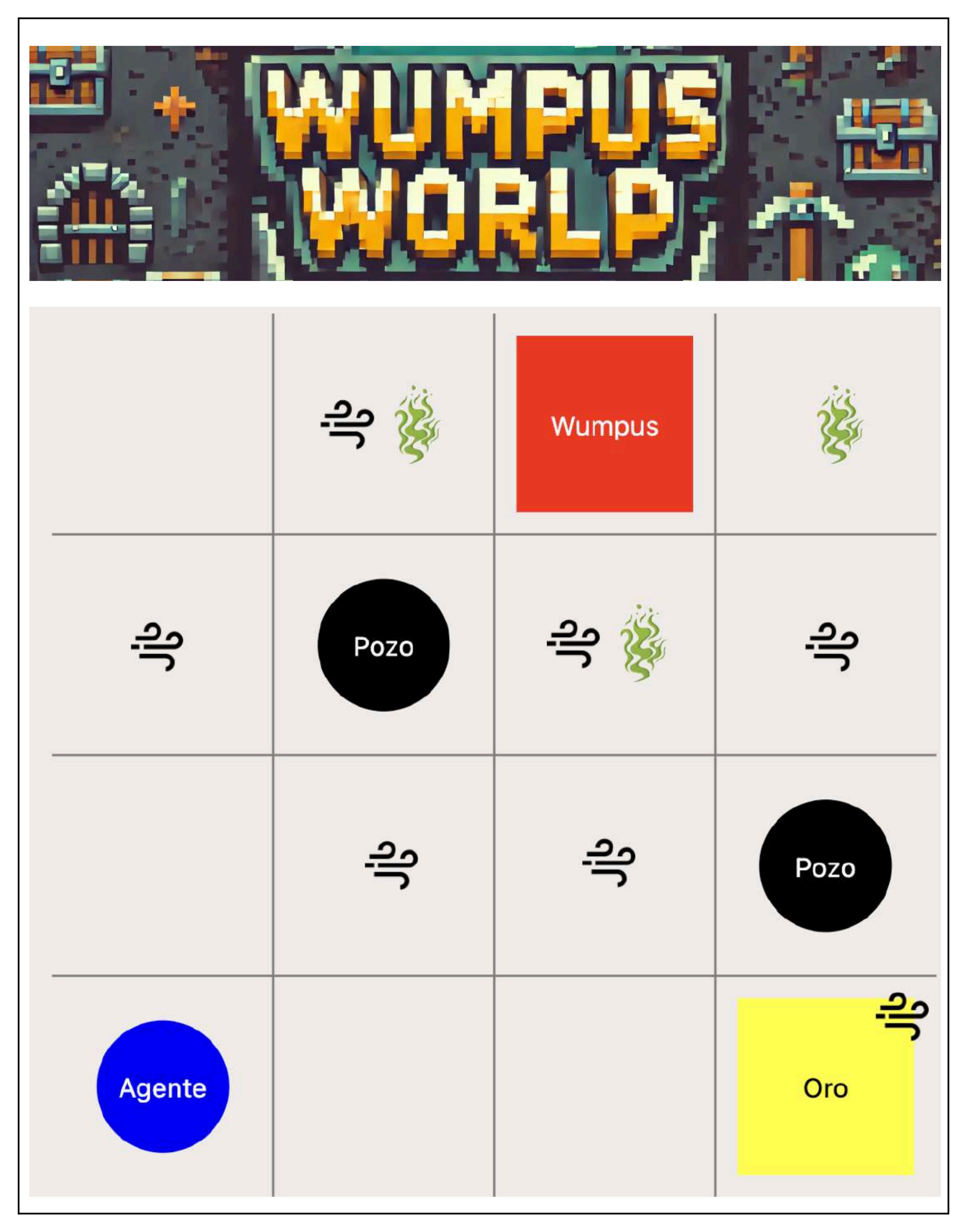


El aprendizaje por refuerzo puede ser descrito como un ciclo de interacción continua entre el agente y el entorno:

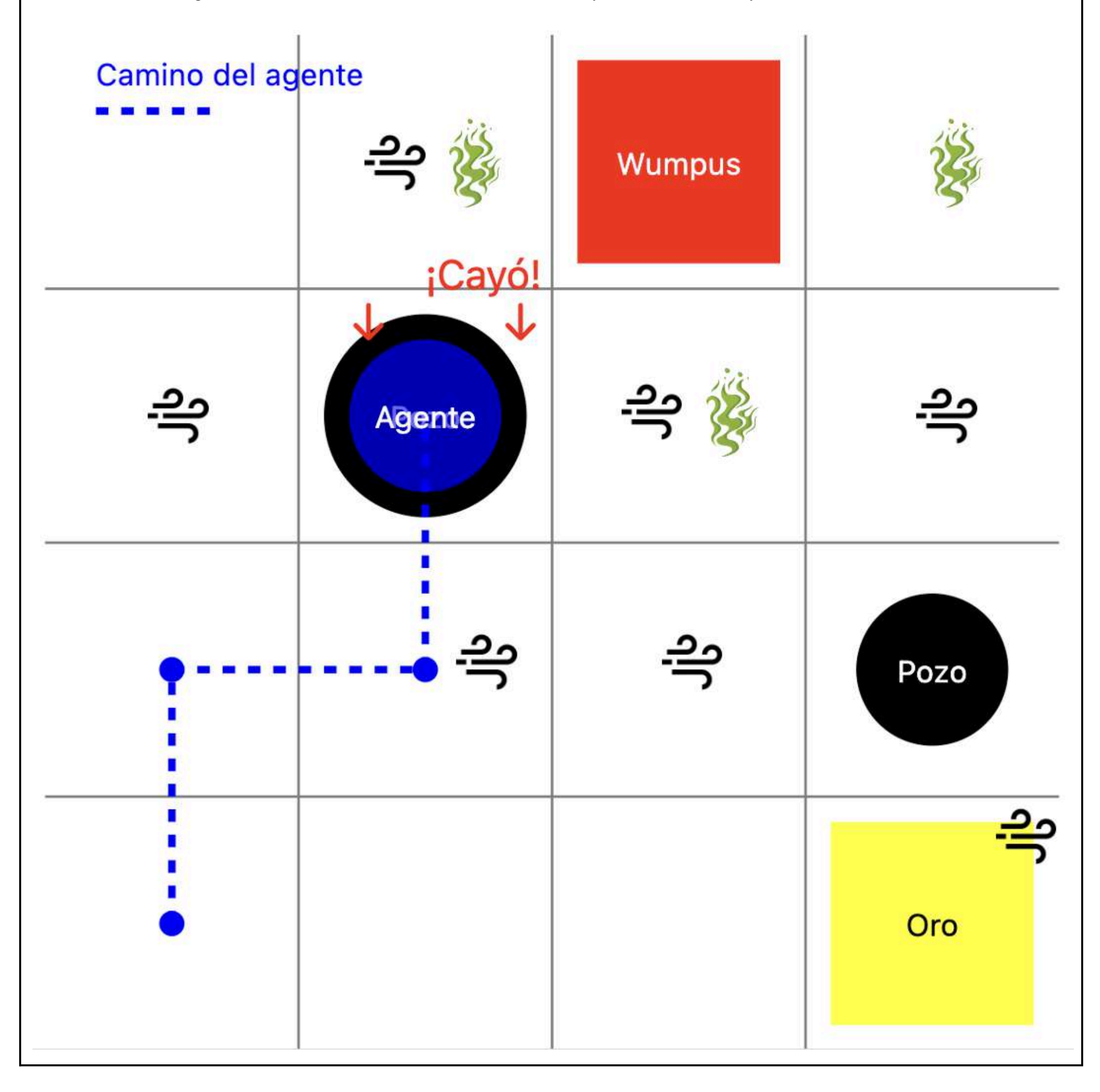
1. 👀 **Percepción del estado**: El agente observa el estado actual del entorno.
2. 🤔 **Selección de acción**: Basándose en su política (estrategia), el agente selecciona una acción.
3. 🎁 **Recepción de recompensa y nuevo estado**: El entorno responde a la acción del agente proporcionando una recompensa y un nuevo estado.
4. 📊 **Actualización de la Política**: El agente ajusta su política para mejorar sus decisiones futuras basándose en la recompensa recibida y el nuevo estado.

Si llegados a este punto te empieza a sonar que esto del *reinforcement learning* se parece mucho a un juego, entonces vamos bien! De hecho esta técnica toma muchos conceptos del mundo de los juegos y videojuegos y podemos verlo en un ejemplo muy famoso, el Wumpus World[54].

[54] El "Wumpus World" fue introducido por Michael Genesereth como un entorno de prueba para inteligencia artificial y fue popularizado por el libro "Artificial Intelligence: A Modern Approach" de Russell y Norvig. Está inspirado en el juego "Hunt the Wumpus", creado por Gregory Yob en 1972. Se ha convertido en un ejemplo clásico para enseñar conceptos de representación del conocimiento y razonamiento en inteligencia artificial

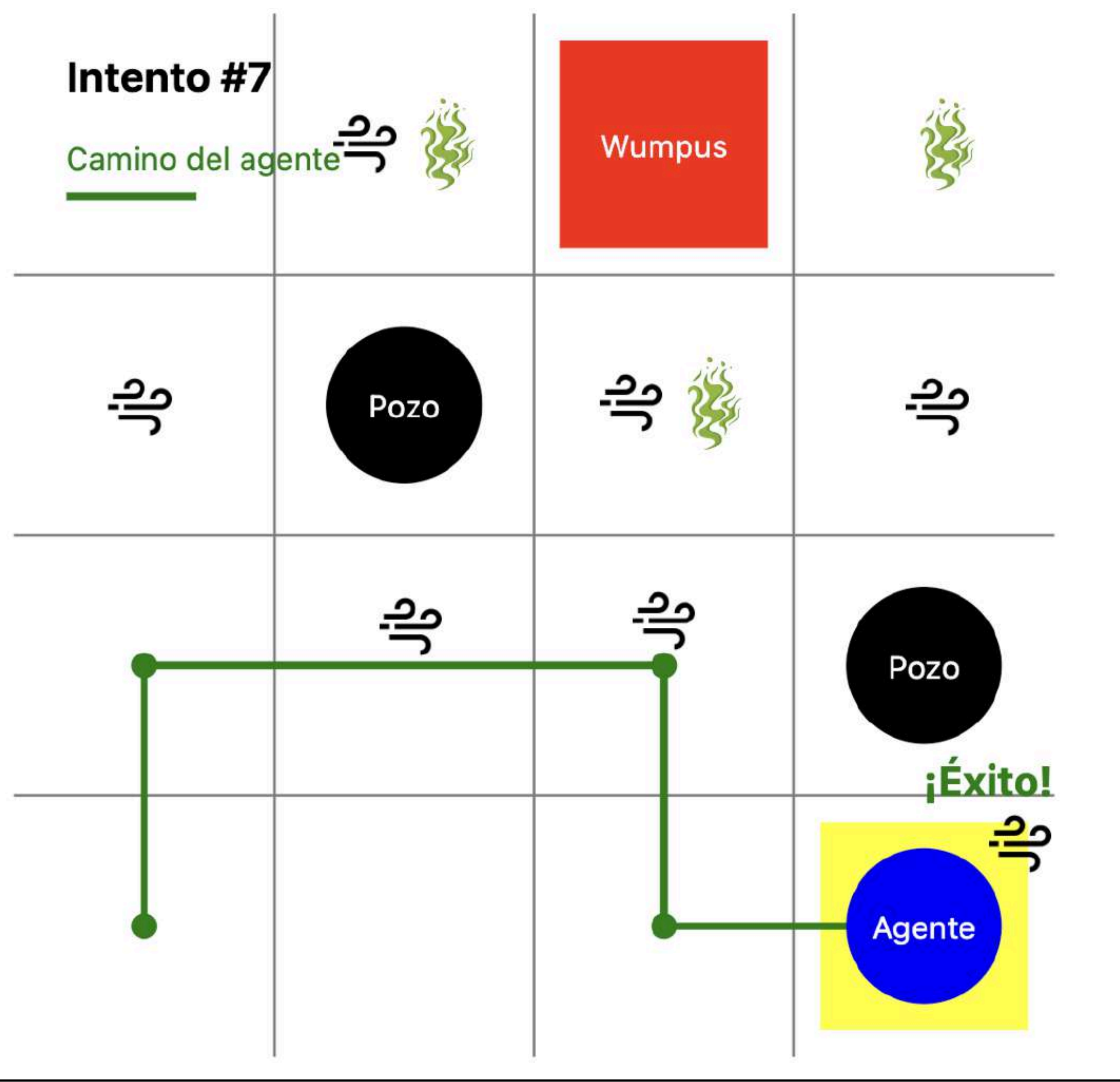

WUMPUS
WORLD
Wumpus
Pozo
Pozo
Agente
Oro

El ***Wumpus World*** es un entorno de aprendizaje clásico en inteligencia artificial, diseñado para ilustrar los principios del aprendizaje por refuerzo y la toma de decisiones en condiciones de incertidumbre. En este mundo, un agente debe navegar por una cuadrícula evitando peligros mortales como pozos y un monstruo llamado Wumpus, mientras busca un tesoro (oro). El agente no puede ver directamente estos elementos, sino que debe inferir su presencia a través de percepciones indirectas como olores[55] y brisas[56]. El objetivo es que el agente aprenda a tomar decisiones óptimas, maximizando su recompensa (encontrar el oro) mientras minimiza los riesgos, utilizando la información limitada disponible en cada paso.



[55] Representado con el símbolo verde que se encuentra adyacente al wumpus. Alerta al jugador acerca de la presencia del wumpus en algún casillero adyacente.
[56] Representado con el simbolo negro que se encuentra adyacente a los pozos. Alerta al jugador acerca de la presencia de un pozo en algún casillero adyacente.

El aprendizaje por refuerzo en el Wumpus World funciona permitiendo que el agente aprenda de sus experiencias a medida que explora el entorno. Inicialmente, el agente no tiene conocimiento sobre la ubicación de los peligros o el oro, por lo que comienza tomando acciones aleatorias. A medida que se mueve por el mundo, el agente percibe señales del entorno: siente una brisa cuando está adyacente a un pozo, y detecta un olor cuando está cerca del Wumpus. Con cada movimiento, recibe *retroalimentación* en forma de recompensas o penalizaciones. Utilizando un algoritmo como Q-learning, el agente actualiza continuamente los valores asociados a cada *par estado-acción* (valores Q), basándose en las recompensas recibidas, las percepciones sensoriales y la estimación de recompensas futuras. Gradualmente, el agente aprende qué acciones son más beneficiosas en cada estado, desarrollando una *política* que le permite interpretar las señales de brisa y hedor para navegar de manera segura hacia el oro mientras evita los peligros. Este proceso iterativo de exploración, aprendizaje y mejora permite al agente optimizar su comportamiento con el tiempo. En el siguiente ejemplo, el agente consiguió en su séptimo intento llegar al oro luego de seis intentos fallidos, aprendiendo a utilizar las percepciones de brisa y hedor para tomar decisiones más informadas.



A continuación, simulamos paso a paso el proceso de aprendizaje de nuestro agente en el Wumpus World, registrando sus decisiones, percepciones y lecciones aprendidas en cada intento. Vamos a poder observar cómo el agente, partiendo de un conocimiento nulo del entorno, va construyendo gradualmente un entendimiento de las señales (brisas y olores) y sus significados, actualizando sus valores Q y refinando su política de acción hasta finalmente descubrir la ruta óptima hacia el oro en su séptimo intento. Los siguientes logs nos permiten ver la evolución del aprendizaje por refuerzo en acción (no necesariamente reflejan la imágen anterior):

[Intento #1]
- Estado inicial: Casilla (1,1)
- Detecto brisa: ¿significado desconocido?
- Acción: Mover arriba
- Resultado: ¡CAÍDA EN POZO! Penalización recibida (-1000)
- Aprendizaje: Brisa = posible pozo cercano. Actualizar valor Q negativo para esta acción

[Intento #2]
- Estado inicial: Casilla (1,1)
- Detecto brisa: Evitar movimiento hacia arriba
- Detecto olor: ¿significado desconocido?
- Acción: Mover derecha
- Resultado: ¡MUERTE POR WUMPUS! Penalización recibida (-1000)
- Aprendizaje: Olor = Wumpus cercano. Actualizar política para evitar casillas con olor fuerte

[Intento #3-5]
- Explorando diferentes rutas pero encontrando peligros
- Refinando mapeo de olores y brisas
- Construyendo matriz de valores Q más precisa
- Identificando patrones de peligro

[Intento #6]
- Estado inicial: Casilla (1,1)
- Mejor comprensión de señales:
  ** Brisa → Pozo adyacente*
  ** Olor → Wumpus cercano*
- Intentó ruta alternativa pero aún subóptima
- Resultado: Caída en pozo pero mejoró mapa mental del entorno

[Intento #7 - ÉXITO]
- Estado inicial: Casilla (1,1)
- Análisis completo de percepciones:
  ** Mapa mental de peligros actualizado*
  ** Valores Q optimizados para ruta segura*
- Ejecución de ruta óptima:
  1. Mover izquierda (evitando pozo conocido)
  2. Mover abajo x2 (zona segura identificada)
  3. Mover derecha (evitando olor del Wumpus)
- ¡ORO ENCONTRADO! Recompensa máxima recibida (+1000)
- Política actualizada: Ruta óptima reforzada para futuros intentos

[Resumen de aprendizaje]
** Tiempo total: 7 iteraciones*

** Política final convergida*
** Mapa de peligros completo*
** Ruta óptima establecida*

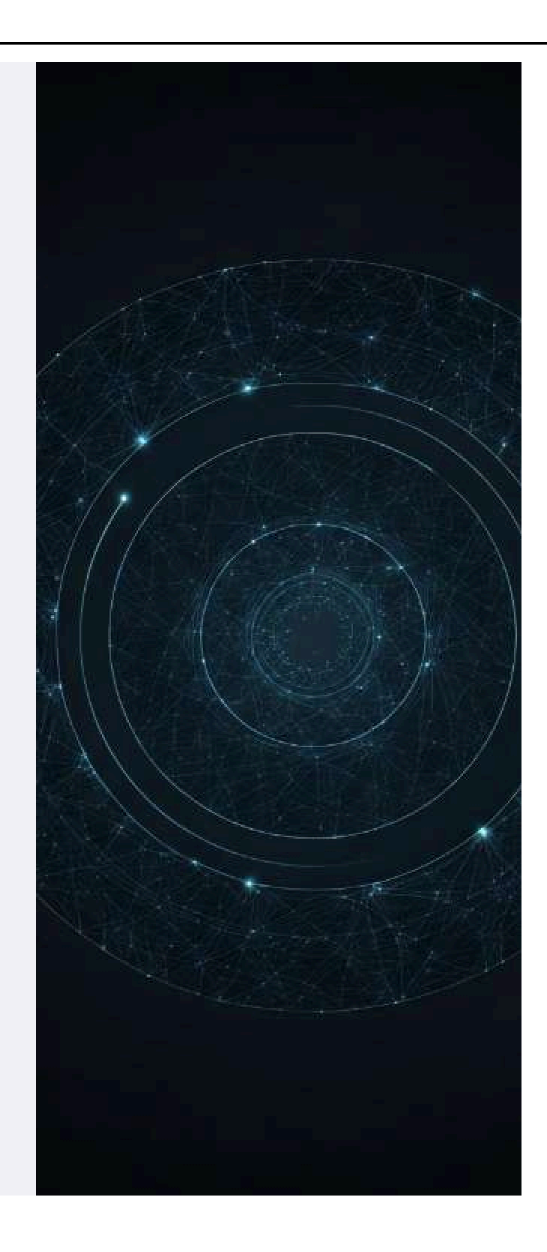
Algoritmos Comunes en Aprendizaje por Refuerzo

No es tan importante que aprendas los algoritmos, pero es bueno que los tengas a mano.

- **Q-Learning**: Un algoritmo que funciona como un "mapa de valores" para cada acción posible en cada situación. Imaginá que tenés una tabla donde anotás qué tan bueno fue hacer cierta acción en cierta situación - si funcionó bien, aumentás su valor; si salió mal, lo disminuís. Con el tiempo, el agente aprende a consultar esta tabla para tomar mejores decisiones, similar a cómo aprendemos de nuestras experiencias pasadas.
- **SARSA (State-Action-Reward-State-Action)**: Es como Q-Learning, pero más "cauteloso". En lugar de asumir que siempre tomarás la mejor acción posible en el futuro, SARSA considera lo que realmente harás. Es como planear un viaje considerando que podrías hacer algunas paradas no planificadas, en vez de asumir que tomarás la ruta más rápida.
- **DQN (Deep Q-Network)**: Imaginá que en lugar de una simple tabla, usas una red neuronal para recordar y generalizar experiencias. Es como pasar de memorizar respuestas específicas a entender patrones generales. Esto es especialmente útil en situaciones complejas, como jugar videojuegos, donde hay demasiadas posibles situaciones como para usar una simple tabla.
- **Política Gradiente**: En vez de evaluar cada acción individualmente, este método ajusta directamente el comportamiento general del agente. Es como ajustar tu estilo de juego completo en lugar de memorizar jugadas específicas. El agente aprende "qué tan bueno" es hacer ciertas acciones en general.
- **Actor-Critic**: Este método combina dos partes: un "actor" que decide qué hacer (como un jugador) y un "crítico" que evalúa qué tan buenas fueron esas decisiones (como un entrenador). El crítico ayuda al actor a mejorar, similar a cómo un entrenador ayuda a un atleta a perfeccionar su técnica proporcionando retroalimentación constante.

El ejemplo del Wumpus World se vincula estrechamente con estos algoritmos de aprendizaje por refuerzo, especialmente con Q-Learning y SARSA. En este entorno, el agente aprende a navegar por el mundo, evitando peligros y buscando el oro, utilizando estos algoritmos para actualizar sus estimaciones del valor de cada acción en cada estado (los valores Q). Por ejemplo, cuando el agente cae en un pozo, recibe una recompensa negativa y actualiza sus valores Q para evitar esa acción en el futuro. Similarmente, cuando encuentra el oro, la alta recompensa refuerza el camino que llevó a ese éxito. Imaginemos el mapa de valores Q en el Wumpus World: al inicio del juego, en la casilla (1,1), moverse hacia la derecha tiene un valor Q inicial bajo de 0.1, mientras que moverse hacia arriba tiene un valor de 0.2. Después de varias exploraciones, si el agente descubre que moverse hacia la derecha desde (1,1) lo lleva cerca del oro, ese valor Q podría aumentar a 0.8, mientras que si detecta una brisa al moverse hacia arriba, ese valor podría disminuir a -0.5. Cada casilla mantiene cuatro valores Q (uno por cada dirección posible) que se van ajustando según las experiencias: valores positivos altos para movimientos que llevan al oro, valores negativos para movimientos que llevan a peligros, y valores cercanos a cero para movimientos poco explorados. A lo largo de múltiples episodios, el agente mejora su política de acción, ilustrando cómo estos algoritmos permiten aprender de la experiencia para optimizar el comportamiento en entornos complejos y con incertidumbre.





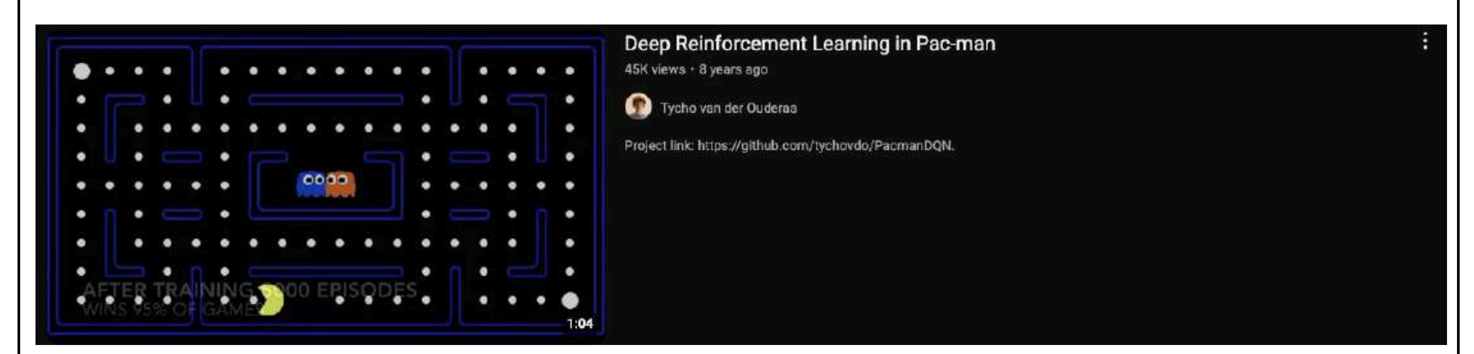
Algunas de las aplicaciones del aprendizaje por refuerzo

- **Juegos**: El aprendizaje por refuerzo ha demostrado ser extremadamente eficaz en el dominio de los juegos, desde escenarios simples como el Wumpus World que acabamos de analizar, hasta juegos de tablero como el ajedrez y el Go, y videojuegos complejos. Si en el Wumpus World el agente aprende a navegar en un entorno discreto con reglas claras y recompensas definidas, en juegos más complejos como el Go, los agentes como AlphaGo de DeepMind deben aprender a evaluar millones de posibles estados del juego y sus consecuencias a largo plazo. Mientras que en el Wumpus World las recompensas son inmediatas (encontrar oro, caer en un pozo), en estos juegos más avanzados el agente debe aprender a valorar movimientos que pueden no tener un beneficio inmediato pero son cruciales para la victoria final.
- **Robótica**: Los robots pueden aprender a realizar tareas complejas mediante la interacción con su entorno y la optimización de sus acciones para maximizar la recompensa, similar a como nuestro agente en el Wumpus World aprende a navegar evitando peligros. Sin embargo, mientras que en el Wumpus World el agente aprende en un entorno discreto y con percepciones simplificadas (brisa, hedor), los robots en el mundo real deben procesar una cantidad masiva de información sensorial continua. Por ejemplo, un robot que aprende a caminar debe coordinar múltiples articulaciones mientras procesa datos de sensores de presión, giroscopios y cámaras, recibiendo recompensas por mantener el equilibrio y avanzar eficientemente. Al igual que el agente del Wumpus aprende de sus

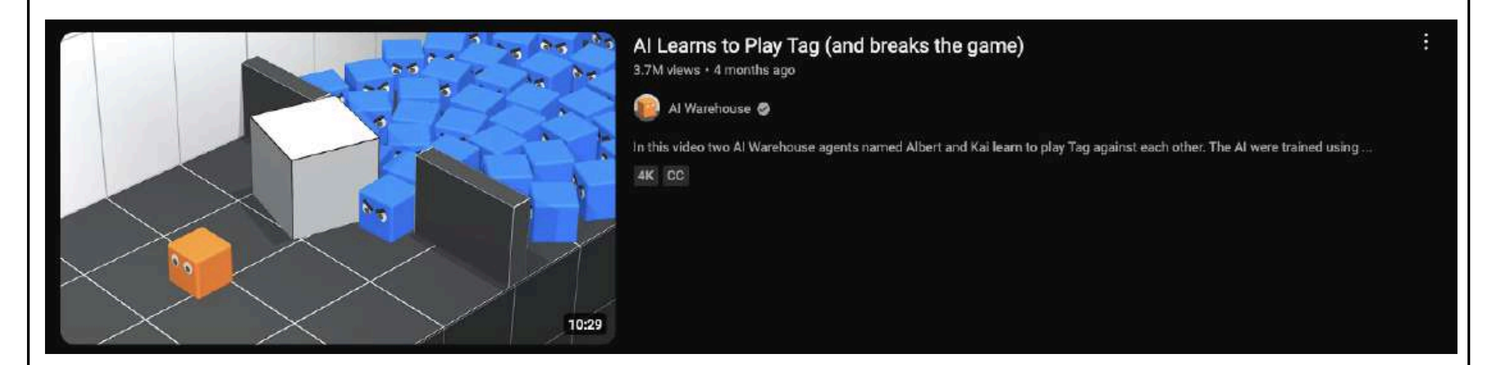
caídas en pozos, estos robots aprenden de sus tropiezos y errores, pero en un entorno físico real con consecuencias más complejas.
- **LLMs**: El aprendizaje por refuerzo también juega un papel crucial en el entrenamiento de modelos de lenguaje como ChatGPT, aunque de una manera más compleja que en el Wumpus World. En este caso, el "entorno" es el contexto de la conversación, las "acciones" son las palabras o tokens que el modelo puede generar, y las "recompensas" vienen de la retroalimentación humana sobre la calidad y utilidad de las respuestas. Por ejemplo, cuando el modelo genera una respuesta útil y precisa, recibe una recompensa positiva; cuando genera contenido inapropiado o incorrecto, recibe una penalización. Esta técnica, conocida como RLHF (Reinforcement Learning from Human Feedback), permite que el modelo aprenda no solo a predecir la siguiente palabra más probable, sino a generar respuestas que sean verdaderamente útiles y alineadas con las preferencias humanas. Al igual que el agente en el Wumpus World aprende a navegar mejor con cada intento, estos modelos aprenden a comunicarse mejor con cada interacción evaluada.

Te dejo dos videos maravillosos para que veas el poder del aprendizaje por refuerzo. El primero es un video muy cortito y el segundo es bastante más largo pero ambos son imperdibles!

Deep Reinforcement Learning in Pac-man



AI Learns to Play Tag (and breaks the game)



Ventajas del Aprendizaje por Refuerzo

- **Adaptabilidad**: Capaz de adaptarse a entornos dinámicos y cambiantes, mejorando continuamente las decisiones del agente basándose en la retroalimentación del entorno. En el Wumpus World, esto se manifiesta cuando el agente aprende a modificar su estrategia si encuentra que ciertos caminos que antes consideraba seguros ahora son peligrosos, o cuando descubre rutas más eficientes hacia el oro. Esta misma capacidad es crucial en aplicaciones del mundo real, como un robot que debe adaptarse a diferentes superficies de terreno o un sistema de trading que debe responder a cambios en el mercado.
- **Capacidad de optimización a largo plazo**: Se enfoca en maximizar la recompensa acumulada a lo

largo del tiempo, lo que es esencial para tareas que requieren planificación y ejecución a largo plazo. Por ejemplo, en el Wumpus World, el agente podría elegir un camino más largo pero más seguro hacia el oro, sacrificando recompensas inmediatas por un mejor resultado final. Este principio se aplica en escenarios como la gestión de recursos energéticos, donde las decisiones actuales deben considerar el impacto futuro en la red eléctrica.
- **Exploración y Explotación**: Balancea entre explorar nuevas acciones y explotar el conocimiento existente para maximizar la recompensa, lo que permite descubrir soluciones óptimas en entornos complejos. En el Wumpus World, esto se refleja cuando el agente debe decidir entre seguir un camino conocido y seguro (explotación) o arriesgarse a explorar una nueva ruta que podría ser más eficiente (exploración). Este balance es fundamental en aplicaciones prácticas, como en sistemas de recomendación que deben decidir entre sugerir contenido que saben que le gusta al usuario o probar con nuevas recomendaciones que podrían expandir sus intereses.

Desafíos del Aprendizaje por Refuerzo

- **Exploración Vs. Explotación**: Este balance es uno de los desafíos fundamentales del aprendizaje por refuerzo. En nuestro ejemplo del Wumpus World, si el agente solo explota los caminos seguros que conoce, podría perderse rutas más eficientes hacia el oro; pero si explora demasiado, podría caer en pozos innecesariamente. Los algoritmos deben implementar estrategias como ε-greedy (donde el agente toma acciones aleatorias con probabilidad ε) o UCB (Upper Confidence Bound) para manejar este dilema. En aplicaciones del mundo real, como en el trading algorítmico, este balance es crucial: explotar demasiado podría significar perderse nuevas oportunidades de mercado, mientras que explorar demasiado podría llevar a pérdidas significativas.
- **Escalabilidad**: Mientras que en el Wumpus World tenemos un espacio de estados relativamente pequeño y acciones discretas, los problemas del mundo real son mucho más complejos. Por ejemplo, un robot que aprende a manipular objetos debe manejar un espacio continuo de estados (posiciones, velocidades, ángulos) y acciones (movimientos precisos de articulaciones). Esto requiere técnicas avanzadas como redes neuronales profundas (DQN) o técnicas de aproximación de función para manejar la explosión combinatoria de estados posibles. En casos extremos, como en el control de una red eléctrica inteligente, el espacio de estados puede ser tan grande que se necesitan arquitecturas especializadas y técnicas de reducción de dimensionalidad.
- **Costo computacional**: El entrenamiento en aprendizaje por refuerzo puede ser extremadamente costoso. Si en el Wumpus World el agente necesita cientos de episodios para aprender una política efectiva, imaginá entrenar un coche autónomo que debe aprender a manejar en todas las condiciones posibles. Esto requiere simuladores sofisticados, infraestructura de computación distribuida y técnicas de optimización avanzadas. Por ejemplo, el entrenamiento de AlphaGo requirió miles de TPUs y millones de juegos de autoaprendizaje.
- **Riesgo de sobreajuste**: Este problema es particularmente crítico en aplicaciones reales. Un agente que aprende perfectamente a jugar el Wumpus World en un mapa específico podría fallar completamente en un mapa diferente. De manera similar, un robot que aprende a caminar en un piso liso podría caerse en terreno irregular. Para combatir esto, se necesitan técnicas como la aleatorización del entorno durante el entrenamiento (domain randomization), el aprendizaje por transferencia, y la exposición a una amplia variedad de escenarios. Por ejemplo, los coches autónomos deben entrenarse en simuladores que incluyan diferentes condiciones climáticas, patrones de tráfico y comportamientos de otros conductores para desarrollar políticas robustas que funcionen en el mundo real.

Algoritmos Comunes en Aprendizaje por Refuerzo

1. **Q-Learning**

   Actualiza valores Q usando recompensa recibida y mejor acción en el siguiente estado.

2. **SARSA**

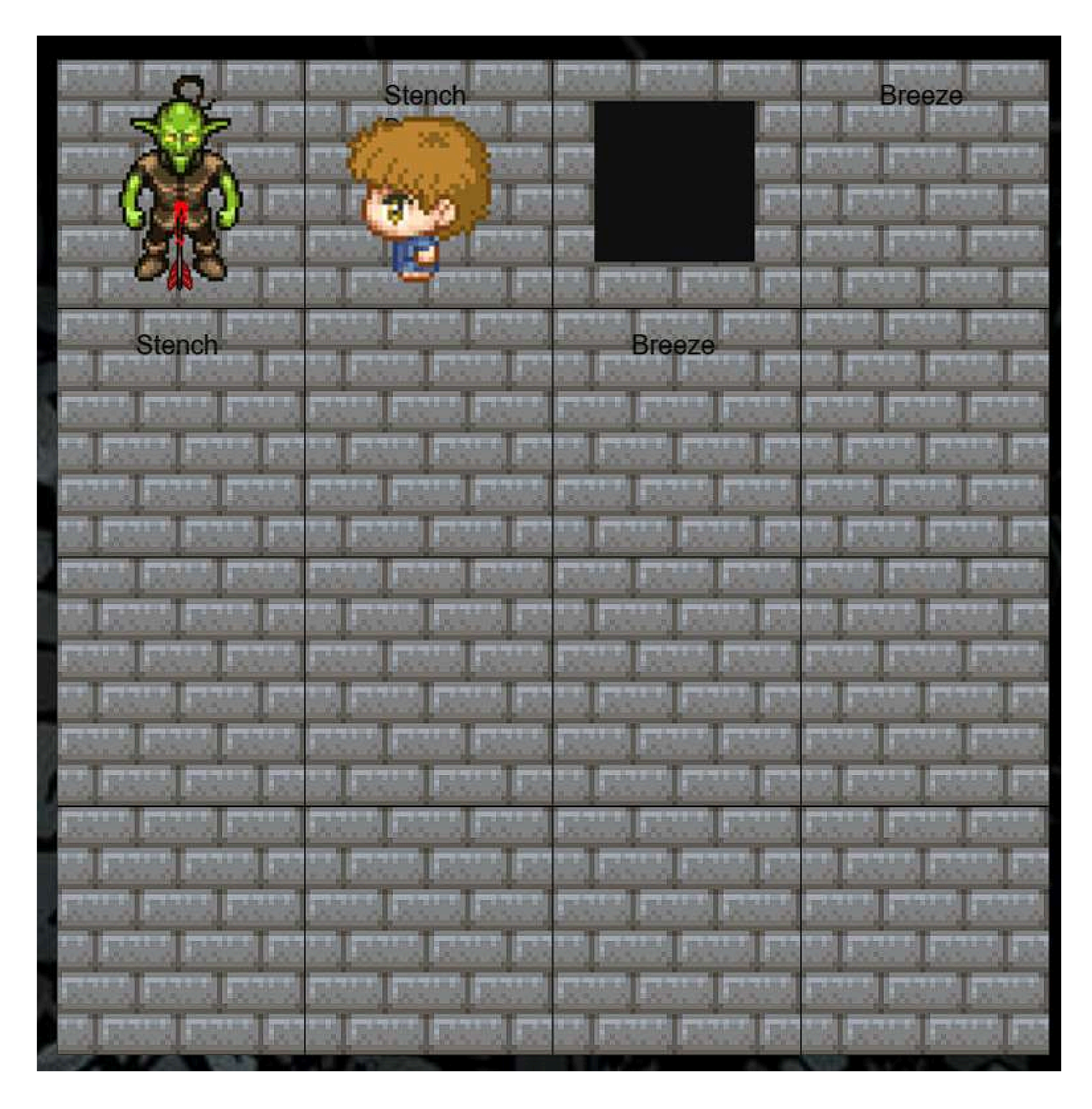
   Actualiza valores Q basándose en la acción realmente tomada en el siguiente estado.

3. **DQN**

   Usa redes neuronales profundas para aproximar la función Q.

¿Querés jugar una versión del Wumpus World?

https://luka1199.github.io/wumpus-world/[57]



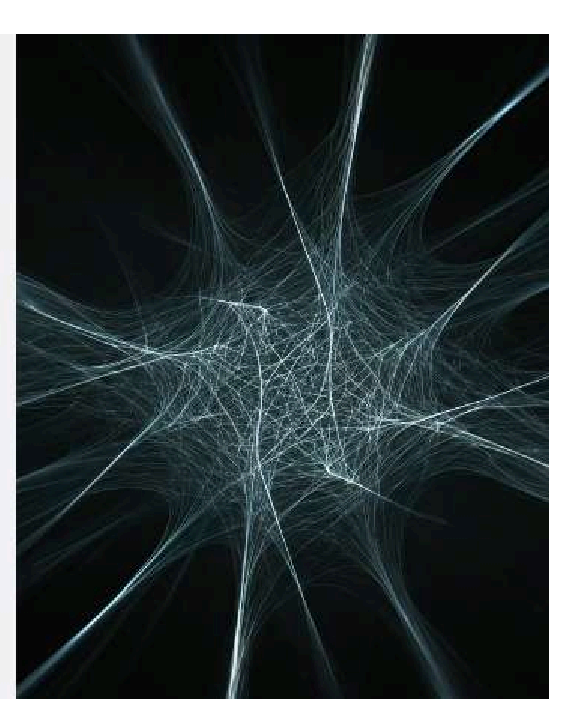
En esta implementación espectacular del usuario de GitHub *luca1199* se puede crear un tablero de diferentes tamaños y explorarlo hasta conseguir vencer al Wumpus. Tené en cuenta que en este caso primero vas a tener que encontrar las flechas para poder usarlas contra el Wumpus y prestar atención a las señales[58] que vas encontrando en el mapa para saber si tenés cerca un pozo o al propio monstruo. Revisá la sección de "Controls" para aprender cómo navegar el tablero y disparar las flechas.

[57] Copyright (c) 2019 Luka Steinbach

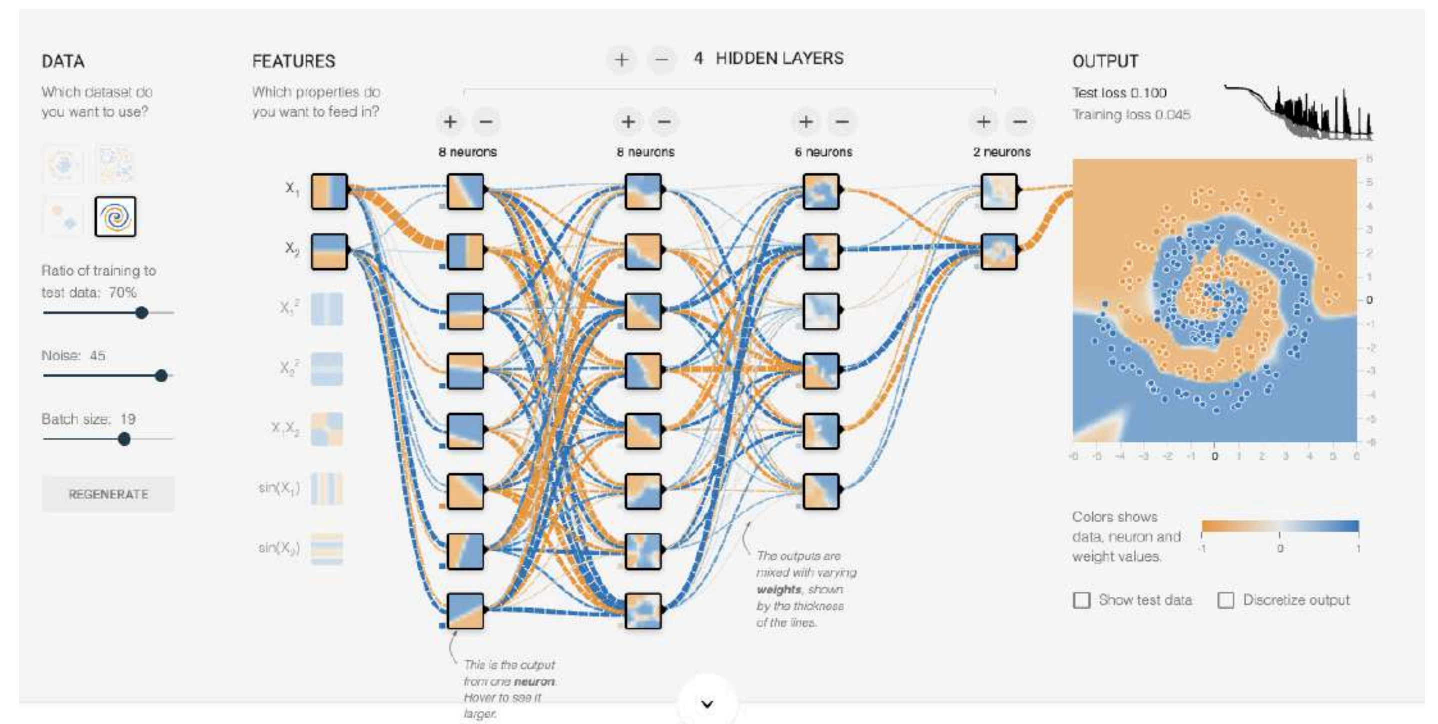
[58] "Breeze" es "Briza". Indica que en un casillero adyacente hay un pozo. "Stench" es "odor/olor". Indica que en un casillero adyacente está el Wumpus.

Redes Neuronales y Deep Learning

Las redes neuronales y el deep learning han revolucionado la IA en la última década. Inspiradas en el cerebro humano, permiten a las máquinas aprender representaciones complejas de datos.



En la sección de algoritmos de aprendizaje supervisado ya pudimos ver una definición sencilla de *redes neuronales*, así que ahora hagamos un ejercicio diferente. Venimos de explorar el mundo del aprendizaje por refuerzo jugando juegos, entonces ¿qué tal si seguimos en este modo interactivo un rato más? Desde hace muchos años, Google tiene una página web extremadamente útil para aprender cómo se entrena una red neuronal. Veámosla :)
https://playground.tensorflow.org/



Mientras vas entrando en el sitio web, recordemos que las redes neuronales son modelos inspirados en el funcionamiento del cerebro humano, donde cada "neurona" artificial procesa y transmite información a otras neuronas en capas sucesivas. El *Deep Learning* lleva este concepto más allá, utilizando múltiples capas de neuronas (de ahí el término "profundo") para aprender representaciones cada vez más abstractas de los datos. Por ejemplo, una red neuronal que aprende a reconocer dígitos escritos a mano primero identifica líneas y curvas simples en las primeras capas, luego combina estas características en formas más complejas en las capas intermedias, hasta finalmente reconocer números completos en las capas finales.

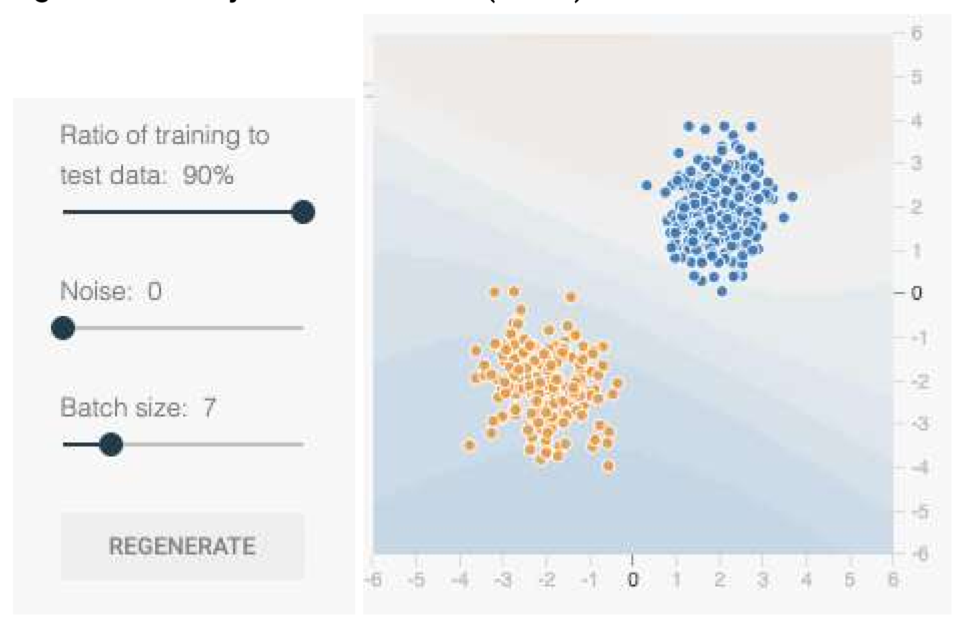
Esta capacidad de aprender automáticamente características cada vez más complejas es lo revolucionó campos como la visión por computadora, el procesamiento del lenguaje natural y el reconocimiento de voz, y es lo que vamos a ver representado en esta brillante simulación. En ella tenés la posibilidad de configurar la arquitectura y prácticamente todos los parámetros típicos de una red neuronal para ver en tiempo real cómo se entrena el modelo aprendiendo de unos datos también simulados. Es una belleza de página y está increíblemente bien ilustrado todo.

Yo solamente me voy a dedicar a dejarte algunos tips de cómo sacarle el máximo provecho. Lo que vos deberías hacer es probar por tu cuenta e ir leyendo esta sección simultáneamente para ir haciendo pequeños cambios y ver sus impactos en la simulación.

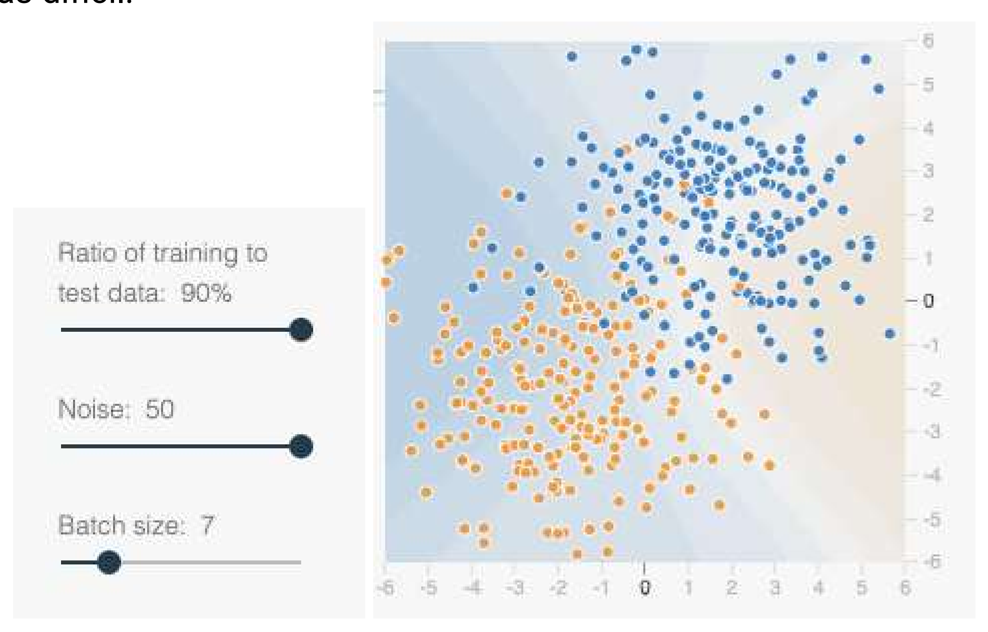


Para empezar, podés primero elegir la *forma* de los datos que el modelo deberá aprender. Son datos simulados en dos dimensiones y no representan ningún caso real específico sino más bien formas claramente escogidas para un ejemplo visual de la versatilidad de los modelos de *deep learning* de tratar con datos de todo tipo de complejidades. En la práctica podemos pensar en estas formas como abstracciones de problemas reales: así como acá vemos círculos y espirales, en aplicaciones reales estos patrones podrían representar la clasificación de correos electrónicos entre spam y no spam, el reconocimiento de diferentes objetos en imágenes, la distinción entre diferentes emociones en texto, o incluso la identificación de patrones en datos médicos. La belleza de esta visualización es que nos permite ver de manera intuitiva cómo una red neuronal aprende a separar y clasificar datos, independientemente de su complejidad subyacente.

También es posible modificar la *separabilidad* de las clases. Podés hacerle el trabajo muy fácil al modelo eligiendo un bajo nivel de *noise* (ruido):

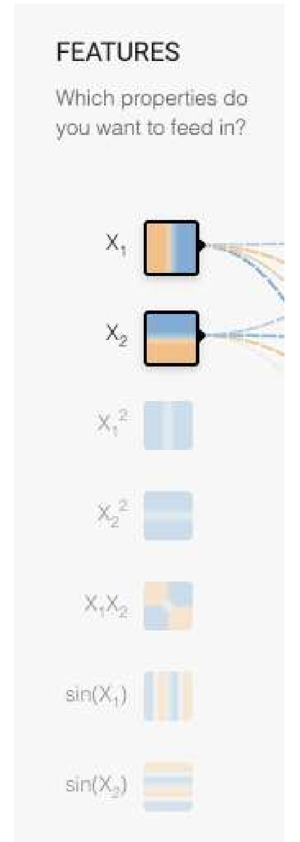


O hacérselo más difícil:



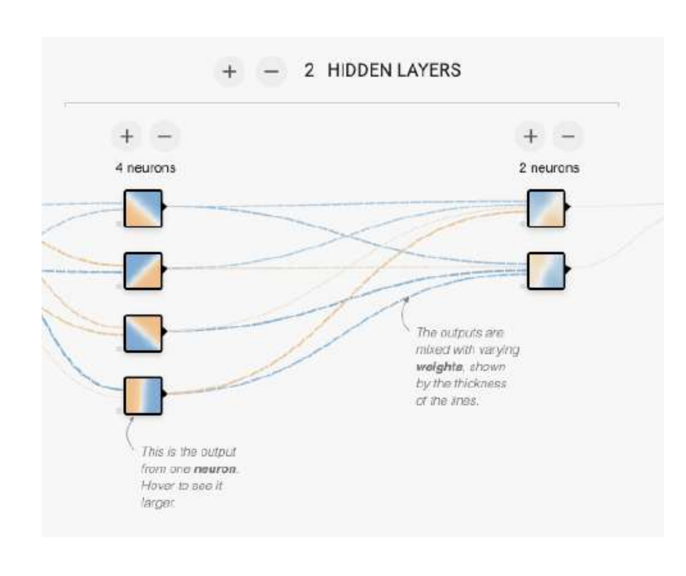
Recordá que *entrenar un modelo de clasificación* se trata de encontrar una función matemática que mejor *separe (clasifique)* las *clases* que queremos ser capaces de predecir. En este caso las *clases* están representadas por los diferentes colores de los puntos, esto deberías pensarlo como imágenes de *gatos vs perros*, por ejemplo. Dos imágenes más arriba, donde *Noise* era 0 podrías pensar como si tuviéramos dos conjuntos de imágenes (gatos y perros) donde claramente se distingue la diferencia de ambos tipos de imágenes, mientras que en el de *Noise* igual a 50, donde hay un solapamiento de puntos naranjas y azules, podrías pensar que algunas fotos son un poco dudosas y podrían tranquilamente confundirse

gatos por perros o viceversa. En ese sentido, entrenar el modelo con *Noise* igual a 50 será más difícil y el modelo cometerá algunos errores con más facilidad.



Continuando con la configuración de esta red neuronal simulada, podés seleccionar qué propiedades utilizar como características de la red. Como se ve en el panel izquierdo de la imagen, tenés opciones que van desde coordenadas simples (X_1, X_2) hasta transformaciones más complejas (X_1^2, X_2^2, X_1X_2) y funciones trigonométricas ($\sin(X_1)$, $\sin(X_2)$). Pensemos en un ejemplo práctico: si estos fueran datos de casas, X_1 podría ser el tamaño y X_2 la antigüedad. Las transformaciones como X_1^2 nos permitirían capturar que el precio no crece linealmente con el tamaño (una mansión vale exponencialmente más), mientras que X_1X_2 podría capturar relaciones combinadas (casas grandes y nuevas tienen un premio extra en el precio). Las funciones trigonométricas podrían capturar patrones cíclicos, como fluctuaciones en el mercado inmobiliario. Esto es un proceso que en general cuando entrenamos un modelo de *deep learning* no se hace de forma manual sino que se deja al modelo aprender, pero en este ejemplo es muy didáctico pensar en cómo ayudar al modelo enseñándole diferentes propiedades que puedan estar relacionadas con nuestros datos. Esto emula el trabajo del científico de datos que debe descubrir *features/características* en los datos para entrenar un modelo con éxito.

Estas propiedades que te ofrece el ejemplo son muy útiles para el tipo de datos con el que estamos jugando, pero imaginate en el caso de imágenes de gatos y perros como si en *features* pudieras tener unos *filtros* que pudieran detectar *orejas de gatos*, u *hocicos de perros*. Ayudaría muchísimo al modelo partir con estas ayudas. Si no las tenés, en la práctica no es muy grave, el modelo puede hacerlo por sí mismo, pero le tomará algo más de tiempo y trabajo. Es justamente esta capacidad de no necesitar demasiado *feature engineering* una de las características más atractivas de los modelos de *deep learning*.



Finalmente, podés elegir la estructura de las *capas intermedias* de la red. Cuantas más capas intermedias y más neuronas, más complejidad y capacidad tendrá el modelo para aprender patrones sofisticados, pero también necesitará más tiempo de entrenamiento y será más costoso computacionalmente. Por ejemplo, una red con una sola capa intermedia de pocas neuronas podría ser suficiente para aprender a separar datos en forma de círculo, pero necesitaríamos más capas y neuronas para aprender patrones más complejos como espirales entrelazadas. Es como construir un edificio: más pisos (capas) y más habitaciones por piso (neuronas) te permiten hacer estructuras más elaboradas, pero requieren más materiales, tiempo y recursos para construir. Este balance entre complejidad y eficiencia es fundamental en el diseño de redes neuronales, y en esta simulación podés experimentar directamente cómo diferentes arquitecturas afectan la capacidad del modelo para aprender los patrones en los datos. Debajo podemos ver un ejemplo de todas las capas y neuronas que podemos disponer en este ejemplo.

Repasando: la imágen anterior es una representación visual de una red neuronal artificial. Se observa una estructura con múltiples capas, específicamente 6 capas ocultas, cada una con 8 neuronas. La red comienza con una capa de entrada que incluye diferentes características (X_1, X_2, X_1^2, X_2^2, X_1X_2, $sin(X_1)$, $sin(X_2)$). Las conexiones entre las neuronas se representan mediante líneas de diferentes colores y grosores, indicando los pesos de las conexiones. A la derecha, se muestra un gráfico de dispersión que representa la salida de la red, con puntos de colores naranja y azul. En la parte superior, se muestran los valores de pérdida (loss) para los conjuntos de entrenamiento y prueba. La interfaz permite ajustar el número de capas y neuronas, así como seleccionar las características de entrada.

Finalmente, una vez estés conforme con la configuración, es cuestión de darle al botón de *Ejecutar* y tu modelo comenzará a entrenarse.



OUTPUT

Test loss 0.085

Training loss 0.104

Epoch son todos los ciclos de entrenamiento y calibración del modelo. Imaginá que estás aprendiendo a cocinar una receta nueva: cada *epoch* es como un intento completo de preparar el plato. El *entrenamiento* es como ir ajustando las cantidades de ingredientes y tiempos de cocción en cada intento, buscando mejorar el resultado. Cada vez que terminás de cocinar (una *epoch*), probás el plato (hacés predicciones) y notás los errores: quizás está muy salado o le falta cocción. Este feedback se "propaga hacia atrás" mediante un proceso llamado *backpropagation*, como si tu cerebro dijera "la próxima vez usá menos sal" o "cocinalo unos minutos más". En cada nuevo intento, vas ajustando estos parámetros gradualmente hasta que el plato sale como querés o hasta que notás que por más que sigas intentando, no mejora más.



La diferencia de escala es fascinante: mientras que en esta simulación de Google el modelo puede entrenarse en segundos al ser una red neuronal simple con pocos parámetros y datos de entrenamiento limitados, un modelo como GPT de OpenAI representa el otro extremo del espectro. Tiene billones de parámetros y una arquitectura de red neuronal extremadamente compleja, lo que hace que su proceso de entrenamiento requiera meses de computación continua en centros de procesamiento masivos. Estos centros de datos están específicamente diseñados y optimizados para el entrenamiento de modelos de IA a gran escala, equipados con miles de GPUs y TPUs trabajando en paralelo, sistemas de enfriamiento especializados y una infraestructura de red de alta velocidad, todo ello necesario para manejar la inmensa cantidad de cálculos requeridos para entrenar estos modelos masivos.

Hermosa demo no? Pero ¿qué tan bien refleja el trabajo real de trabajar con redes neuronales en la práctica?

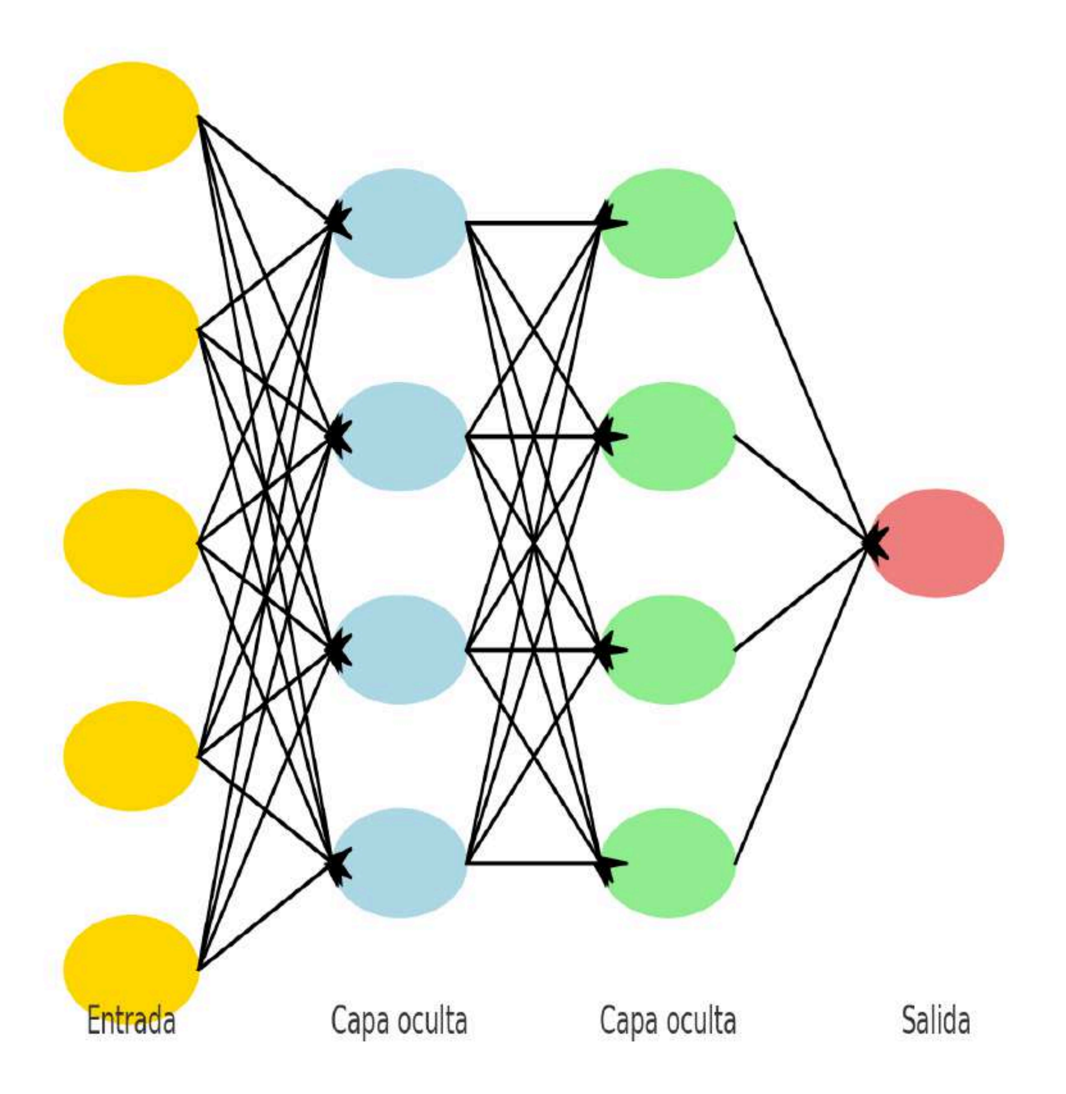
Te sorprenderá saber que esta simple demostración refleja con bastante precisión el proceso real de trabajar con redes neuronales. Al igual que en esta simulación, donde ajustamos capas, neuronas y características mientras observamos en tiempo real cómo el modelo aprende, en el mundo real nuestra tarea principal es exactamente esa: encontrar la mejor configuración del modelo para *ayudarle* a completar su tarea de forma satisfactoria y en el menor tiempo posible. Por lo general comenzamos trabajando con una arquitectura estándar y vamos introduciendo mejoras poco a poco a medida que vemos cómo responde el modelo a los datos de entrenamiento. Considero que entrenar un modelo de *deep learning* es uno de los trabajos más *artesanales* de un ingeniero de IA o un científico de datos - es un proceso de

prueba y error, de intuición y experiencia, de ajustar parámetros y observar resultados, muy similar a lo que acabás de experimentar en esta demo.

Espero que te guste esta demo, a mi ciertamente me ayudó mucho en su momento (si! lleva años ahí) y sigue siendo una de las mejores herramientas para entender visualmente cómo funciona el proceso de entrenamiento de una red neuronal.

Ahora sí, luego de *aprender jugando*, pasemos a definiciones más formales de las redes neuronales y exploremos sus subtipos más emblemáticos:

Inspiradas en la estructura y el funcionamiento del cerebro humano, las redes neuronales permiten a las máquinas aprender representaciones complejas y jerárquicas de los datos. Las redes neuronales y el *deep learning* son la base de muchos de los avances recientes en visión por computadora, procesamiento de lenguaje natural y otras áreas clave de la IA.



- 🧠 Las redes neuronales artificiales consisten en neuronas artificiales organizadas en capas.

- ⚡ Cada neurona recibe entradas, las procesa y produce una salida.
- 🔢 Las capas incluyen una **capa de entrada,** una o más **capas ocultas** y una **capa de salida**.
- 🕵️ Las **capas ocultas** permiten la abstracción de características a diferentes niveles.
- ⚖️ Los **pesos y biases** de las conexiones entre neuronas se ajustan durante el entrenamiento para minimizar la función de pérdida, que cuantifica el error entre las predicciones del modelo y las etiquetas reales. Es como ajustar los controles de una mesa de mezclas de sonido: cada peso es como un potenciómetro que necesitamos calibrar precisamente para conseguir el mejor sonido (en nuestro caso, las mejores predicciones).
- 📊 Las **funciones de activación**, como la sigmoide, tangente hiperbólica (tanh) y ReLU (Unidad Lineal Rectificada), ayudan a las neuronas a introducir no linealidades, permitiendo que la red capture patrones complejos. Podemos pensar en las funciones de activación como en interruptores inteligentes que deciden cuándo y cuánto debe "encenderse" cada neurona: algunos son como interruptores suaves (sigmoide) que se encienden gradualmente, otros son más bruscos como ReLU que es similar a un interruptor de luz común.
- 🔄 El **proceso de aprendizaje** en una red neuronal implica la propagación hacia adelante, donde los datos se pasan a través de la red para calcular las salidas, y la propagación hacia atrás (*backpropagation*), donde los pesos se ajustan para reducir el error basado en las diferencias entre las predicciones y los valores reales. Es similar a cómo aprendemos a cocinar: primero seguimos una receta (propagación hacia adelante), probamos el resultado, y luego ajustamos los ingredientes según qué tan bien salió (propagación hacia atrás).
- 🔧 Los **algoritmos de optimización**, como el *descenso de gradiente estocástico* (SGD), Adam y RMSprop, se utilizan para ajustar los pesos y biases de manera eficiente. Es como bajar una montaña en bicicleta buscando el camino más eficiente: SGD sería como bajar directamente siguiendo la pendiente más pronunciada, mientras que algoritmos más sofisticados como Adam serían como tener un GPS que nos ayuda a encontrar atajos y evitar obstáculos.

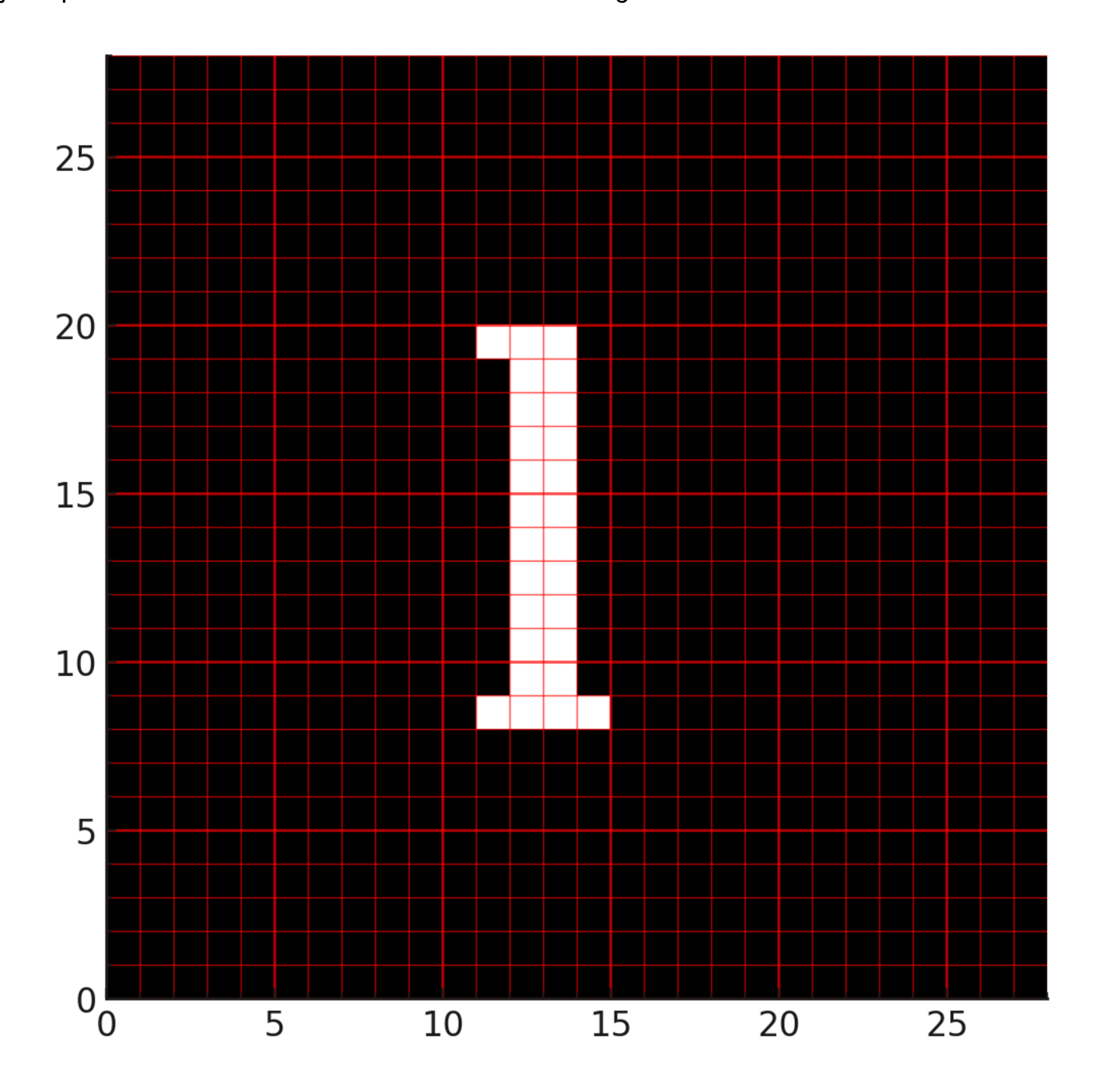
__Tipos de Redes Neuronales__

- Las **redes neuronales artificiales (ANN, por *artificial neural networks*)** son la forma más básica y se utilizan para una variedad de tareas de clasificación y regresión. Son modelos versátiles que se pueden aplicar a diversos problemas de predicción y clasificación. Acá hay ejemplos concretos de aplicaciones:
    - **Regresión:**
        - 📈 **Finanzas**: Predecir precios de acciones, riesgo crediticio o demanda de productos
        - 🌦️ **Clima**: Pronóstico del tiempo, predicción de temperaturas o precipitaciones
        - 🏭 **Industria**: Estimación del desgaste de maquinaria, consumo energético
        - 🛒 **Ventas**: Predicción de ventas futuras, optimización de inventario
    - **Clasificación:**
        - 🏥 **Medicina**: Detectar enfermedades en imágenes médicas, clasificar células
        - 🎯 **Marketing**: Segmentación de clientes, identificar clientes potenciales
        - 🔒 **Seguridad**: Detección de fraude en transacciones bancarias
        - 💻 **Tecnología**: Filtrado de spam, reconocimiento de voz, clasificación de documentos

    La ventaja principal de las ANN es su capacidad para aprender patrones complejos de los datos, incluso cuando la relación entre las variables no es obvia o lineal. Sin embargo, requieren una cantidad significativa de datos para entrenarse efectivamente.

- Las **redes neuronales convolucionales (CNN)** son especialistas en procesamiento de datos con una

estructura de grilla, como imágenes, y utilizan *convoluciones*[59] para extraer características espaciales jerárquicas. Veremos esto en detalle en una de las siguientes secciones.



En esta imagen podemos ver cómo un simple número "1" está representado en una grilla de píxeles. Cada cuadrado en la grilla representa un píxel individual, que puede estar "encendido" (blanco) o "apagado" (negro). Esta representación ilustra el concepto fundamental de que cualquier imagen digital, por compleja que sea, no es más que una matriz de píxeles, donde cada píxel contiene información sobre su color o intensidad. En este caso simple en blanco y negro, cada píxel solo puede tener uno de dos valores posibles, pero en imágenes más complejas, cada píxel podría contener información sobre diferentes niveles de rojo, verde y azul (RGB). Este tipo de datos estructurados en forma de grilla son típicamente procesados por Redes Neuronales Convolucionales (CNN), que funcionan de manera similar a cómo nuestro cerebro procesa la información visual: primero detectando características simples (como bordes o esquinas) y luego combinándolas para reconocer patrones más

[59] Una convolución, en el contexto de las redes neuronales, es como usar una lupa especial que se mueve sobre una imagen. Esta "lupa" (llamada filtro o kernel) examina pequeñas secciones de la imagen a la vez, buscando patrones específicos como bordes, texturas o formas. A medida que se desliza sobre la imagen, combina la información de cada área pequeña para crear un nuevo "mapa" que resalta características importantes. Es similar a cómo nuestros ojos pueden enfocarse en detalles específicos de una escena para entender mejor lo que estamos viendo. Este proceso permite a la red neuronal aprender y reconocer características clave en imágenes de manera eficiente.

complejos (como formas, texturas y finalmente objetos completos).

- Las **redes neuronales recurrentes (RNN)** son modelos especializados en procesar información secuencial, como si fueran lectores que recuerdan el contexto de lo que han leído anteriormente. Podemos entenderlas mejor con ejemplos prácticos:

  RNN Básica:

  - 📝 **Texto**: Predecir la siguiente palabra en una oración basándose en las palabras anteriores. Por ejemplo, al ver "Me gusta tomar café con..." la red podría predecir "leche" o "azúcar" basándose en patrones comunes en el lenguaje. Antes de la revolución de los modelos transformers y GPT, las RNN, especialmente en su variante LSTM (Long Short-Term Memory), eran el estándar de oro para el procesamiento de lenguaje natural y la generación de texto. Su capacidad para "recordar" información relevante a lo largo de secuencias las hacía ideales para tareas como completar texto, traducción automática y análisis de sentimiento. Sin embargo, tenían limitaciones para manejar dependencias a largo plazo y para el procesamiento en paralelo, problemas que los transformers posteriormente resolverían.
  - 📊 **Finanzas**: Analizar tendencias en precios de acciones considerando no solo el precio actual sino también su evolución histórica, lo que permite detectar patrones temporales en el mercado. Por ejemplo, la red podría aprender que después de una caída brusca seguida de tres días de recuperación lenta, frecuentemente sigue un período de estabilidad.
  - 🎵 **Música**: Generar secuencias de notas musicales que sigan un patrón coherente, como si la red hubiera aprendido el "estilo" de diferentes compositores y pudiera crear melodías similares. Por ejemplo, después de analizar las composiciones de Bach, la red podría aprender que después de una secuencia específica de notas en una fuga, típicamente sigue un patrón de contrapunto característico.
  - 🌡️ **Clima**: Predecir temperaturas basándose en datos históricos, teniendo en cuenta no solo las mediciones recientes sino también patrones estacionales y tendencias a largo plazo. Por ejemplo, la red podría aprender que después de tres días de temperaturas ascendentes en primavera seguidos de un aumento en la humedad, frecuentemente sigue un día de lluvia.

  🧠 **Variantes Avanzadas**:

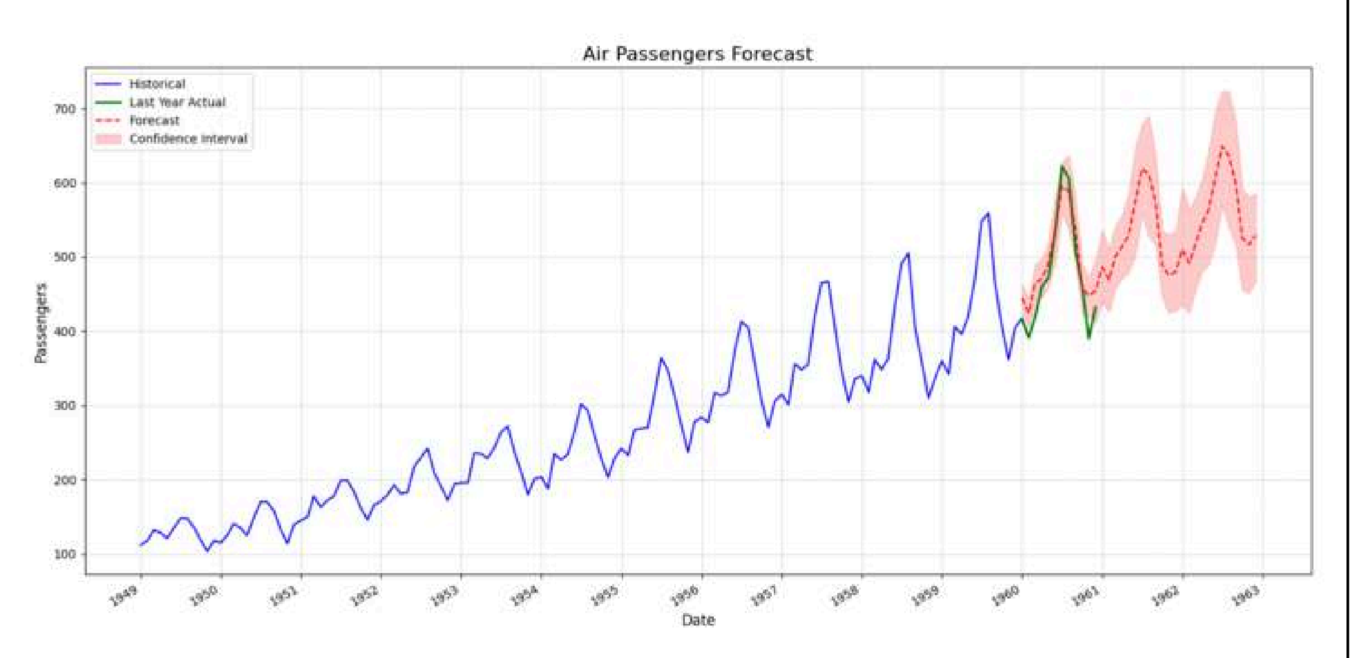
  - **LSTM** (Long Short-Term Memory): estas redes resuelven el problema de la "memoria de largo plazo" que tienen las RNN básicas mediante un sofisticado sistema de "puertas" que controlan qué información se debe recordar y qué olvidar.
    - 📚 Como un lector que puede recordar información importante de capítulos anteriores mientras descarta detalles irrelevantes. Por ejemplo, al leer una novela, puede mantener en memoria quiénes son los personajes principales y sus motivaciones, aunque hayan aparecido muchos capítulos atrás.
    - 🗣️ Útil en traducción de idiomas porque puede mantener el contexto y significado a lo largo de oraciones largas. Especialmente importante en idiomas donde el orden de las palabras es muy diferente, como entre el inglés y el japonés.
    - 📱 Reconocimiento de voz en asistentes virtuales, donde necesita recordar el inicio de una frase para entender correctamente su significado completo, como en "Envía un mensaje a... [pausa larga]... Juan".
  - **GRU** (Gated Recurrent Unit): una variante más moderna y simplificada de LSTM que combina algunas de sus puertas para lograr un rendimiento similar con menor complejidad computacional.
    - 🚀 Versión más simple y rápida que LSTM porque usa menos "puertas" para controlar el flujo de información. Es como una versión optimizada que sacrifica algo de flexibilidad por eficiencia computacional.
    - 📈 Predicción de series temporales donde las relaciones no son demasiado

complejas o distantes en el tiempo, como predicción de consumo eléctrico o tráfico web.
   - 🤖 Chatbots y procesamiento de lenguaje natural para conversaciones cortas donde no se necesita mantener contexto durante largos períodos, como sistemas de atención al cliente automatizados.

La ventaja de ambos enfoques principalmente es su "memoria": pueden recordar información relevante por largos períodos, como entender el contexto en una conversación o patrones en una serie temporal.

Efectividad para series temporales

Una serie temporal es una secuencia de datos ordenados cronológicamente, como registros diarios de temperatura, mediciones mensuales de ventas, o en el caso del siguiente gráfico, el número de pasajeros aéreos a lo largo del tiempo. Tradicionalmente, estas series se han analizado con modelos estadísticos clásicos como ARIMA, SARIMA o modelos de suavizado exponencial, que son efectivos para capturar patrones estacionales y tendencias. Sin embargo, con el aumento en la complejidad y volumen de los datos, los modelos basados en redes neuronales están ganando popularidad por su capacidad para capturar patrones no lineales y relaciones más complejas en los datos. Las Redes Neuronales Recurrentes (RNN) son especialmente efectivas para analizar series temporales, como muestra este gráfico de predicción de pasajeros aéreos. El gráfico representa datos históricos de tráfico de pasajeros aéreos (línea azul), junto con datos del último año (línea verde) y una predicción futura (línea roja punteada), rodeada por un intervalo de confianza (área sombreada rosa) que indica el rango de incertidumbre en la predicción.



Las RNN son particularmente adecuadas para este tipo de datos secuenciales porque pueden mantener una "memoria" de observaciones anteriores, permitiéndoles capturar tanto tendencias a largo plazo como patrones estacionales recurrentes. Esta capacidad de recordar y aprender de secuencias temporales las hace ideales para predecir series de tiempo, ya sean patrones de tráfico aéreo, precios de acciones, demanda de productos o cualquier otro fenómeno que evolucione a lo largo del tiempo con dependencias temporales.

- Las **redes generativas adversarias (GAN)** constan de un *generador* y un *discriminador* que compiten entre sí, utilizadas para generar datos sintéticos que parecen reales. Funcionan como un equipo de

dos redes neuronales que compiten entre sí en un proceso similar a un artista y un crítico de arte. Antes del surgimiento de los modelos de difusión como DALL-E o Stable Diffusion, las GAN dominaron el campo de la generación de imágenes durante varios años, produciendo resultados impresionantes en aplicaciones como la generación de rostros de personas que no existen, la conversión de bocetos en imágenes fotorrealistas, o la restauración de fotos antiguas. Aunque los modelos de difusión las han superado en la generación de imágenes a partir de texto, las GAN siguen siendo muy útiles en casos específicos como la generación de datos sintéticos para entrenamiento, la traducción entre dominios (como convertir fotos de día a noche), y en aplicaciones donde se requiere generación en tiempo real, ya que suelen ser más rápidas que los modelos de difusión.



El proceso de las GAN:

🎨 **El Generador** (El Artista):

- Intenta crear "obras" (datos sintéticos) cada vez más realistas
- Aprende de las críticas para mejorar sus creaciones
- Su objetivo es "engañar" al crítico

🔍 **El Discriminador** (El Crítico):

- Examina tanto obras reales como generadas
- Intenta distinguir lo real de lo artificial
- Su objetivo es no ser engañado

⚔️ **La Competencia**:

- Cuanto más aprende el crítico a detectar falsificaciones, más se esfuerza el artista en mejorar
- Con el tiempo, las creaciones se vuelven tan buenas que son difíciles de distinguir de las reales

🎯 **Aplicaciones Prácticas**:

- 🖼️ Generar imágenes realistas
- 👤 Crear rostros de personas que no existen
- 🎮 Mejorar gráficos de videojuegos
- 🎨 Convertir bocetos en imágenes realistas
- 📷 Restaurar fotos antiguas



Esta imagen muestra un ejemplo del poder de las GANs en la restauración y colorización de fotografías históricas. A la izquierda vemos una fotografía original en blanco y negro de dos bailarines de ballet de finales del siglo XIX, y a la derecha el resultado después de que una GAN ha añadido color de forma realista. La red ha aprendido a inferir los colores apropiados para diferentes elementos como el tutú, los trajes y el fondo, basándose en su entrenamiento con miles de pares de imágenes históricas y sus versiones modernas colorizadas. Este tipo de aplicación es particularmente valiosa en la preservación y restauración del patrimonio histórico fotográfico.

Durante la pandemia del año 2020 dediqué meses a estudiar las GANs aplicadas a todo tipo de casos de uso. El más divertido, la transferencia de estilos, donde tenemos una imágen a la que queremos aplicarle un estilo en particular, como en el siguiente ejemplo[60]:

[60] https://www.tensorflow.org/tutorials/generative/style_transfer





Esta imagen demuestra la aplicación de transferencia de estilo usando GANs, donde se combina el contenido de una imagen (un Labrador) con el estilo artístico de otra (una pintura abstracta de Kandinsky). En la parte superior vemos las imágenes de entrada: la foto original del perro y la obra de arte que proporciona el estilo. Abajo, el resultado de la GAN que ha "repintado" al perro manteniendo su forma y estructura pero adoptando los patrones coloridos y pinceladas características del estilo artístico abstracto. Esta técnica permite crear nuevas obras artísticas que preservan el contenido de una imagen mientras adoptan las características estilísticas de otra.

Las GANs, como ya hemos dicho, han cedido protagonismo en el ámbito público desde la irrupción de los modelos de difusión, que han alcanzado resultados espectaculares en la generación de imágenes. Mientras que las GANs fueron revolucionarias en su momento, logrando hitos como la generación de rostros fotorrealistas con StyleGAN, los modelos de difusión como DALL-E, Stable Diffusion y Midjourney han establecido un nuevo estándar en la generación de imágenes, especialmente en la capacidad de crear contenido visual a partir de descripciones textuales con un nivel de detalle y coherencia sin precedentes. Sin embargo, las GANs siguen siendo relevantes en aplicaciones específicas donde sus características únicas son ventajosas, como en la generación en tiempo real, la traducción entre dominios visuales y la creación de datos sintéticos para entrenamiento.

- Estos **modelos de difusión** que han tomado el liderazgo desde 2020 funcionan de manera fundamentalmente diferente: en lugar de usar dos redes que compiten entre sí (como las GANs) o comprimir información (como los Autoencoders), utilizan un proceso inspirado en la física que se asemeja a cómo un artista restauraría una imagen dañada. Mediante una arquitectura especial llamada U-Net, aprenden a revertir paso a paso un proceso de degradación, como si fueran reconstruyendo una imagen que se ha ido deteriorando gradualmente. Esta aproximación ha demostrado ser notablemente más estable y efectiva que las arquitecturas anteriores, especialmente cuando se combina con el poder de los modelos de lenguaje grandes para interpretar descripciones textuales.

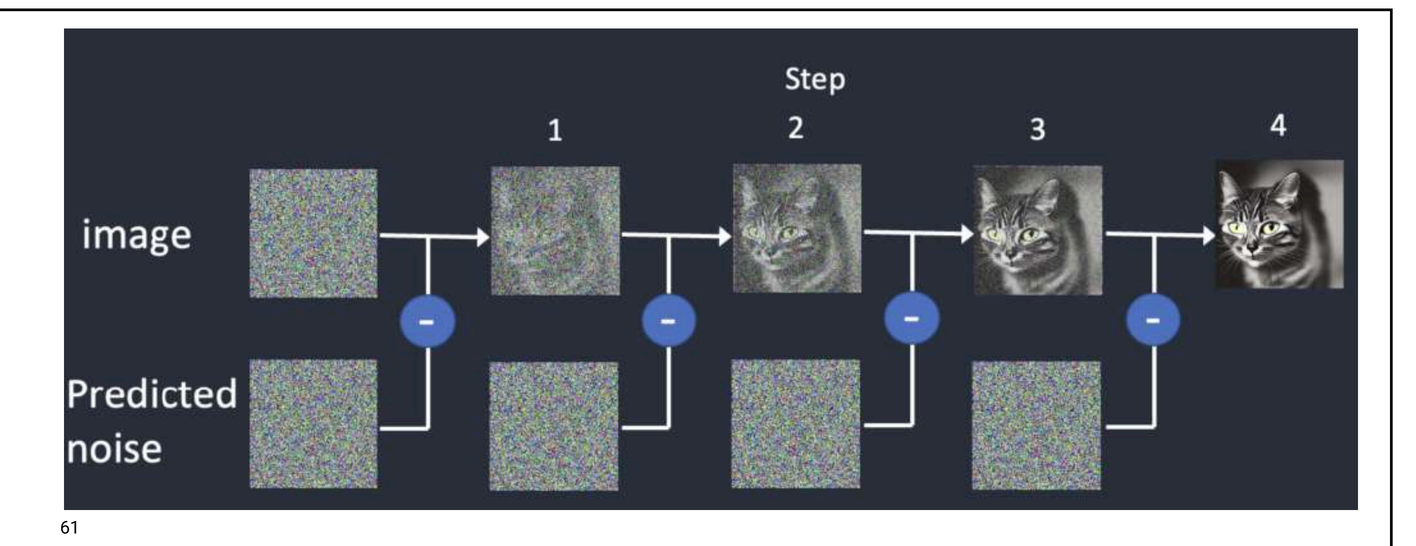
  El proceso funciona en tres fases:

  1. 🌫️ **Destrucción Gradual**: Parte de una imagen clara y añade ruido progresivamente hasta convertirla en ruido aleatorio
  2. 🧠 **Aprendizaje**: La red aprende el proceso inverso:
     - Cómo eliminar el ruido paso a paso
     - Qué detalles recuperar en cada etapa
     - Cómo refinar la imagen gradualmente
  3. ✨ **Generación**: Comienza con ruido aleatorio y aplica el proceso inverso aprendido, refinando la imagen paso a paso hasta crear algo coherente

  Servicios Populares:

  - 🎨 **DALL-E** (OpenAI): Pionero en generación de imágenes desde texto
  - 🖼️ **Midjourney**: Famoso por su estética artística y resultados creativos
  - ⚡ **Stable Diffusion**: Modelo de código abierto, altamente personalizable

  Esta arquitectura resultó ser más estable y fácil de entrenar que sus predecesoras, permitiendo generar imágenes sorprendentemente realistas a partir de descripciones textuales, mejorar imágenes existentes o completar partes faltantes.



[61]

Esta imagen ilustra el proceso de un modelo de difusión trabajando en reversa, desde el ruido hasta una imagen clara de un gato. En la fila superior, vemos cómo la imagen evoluciona a través de varios pasos, comenzando con ruido aleatorio (paso 1) y gradualmente revelando formas más definidas hasta llegar a una imagen nítida de un gato (paso 4). En la fila inferior, vemos el "ruido predicho" que el modelo va eliminando en cada paso. Este proceso es similar a un artista que comienza con un lienzo caótico y va refinando gradualmente los detalles hasta obtener una imagen clara. La clave del éxito de los modelos de difusión radica en su capacidad para aprender exactamente cuánto ruido eliminar en cada paso para revelar gradualmente la imagen deseada.

- **Autoencoders**: Mientras los modelos de difusión y las GANs se centran principalmente en la generación de imágenes nuevas, existe otro tipo de arquitectura neural que aborda el problema desde una perspectiva diferente Los **autoencoders** son redes diseñadas para aprender una representación codificada eficiente de los datos, y son usados para tareas como la reducción de dimensionalidad y la detección de anomalías. Son como un experto en compresión de imágenes que aprende a preservar los detalles más importantes mientras reduce significativamente el tamaño del archivo. Imaginemos el proceso con una fotografía digital:

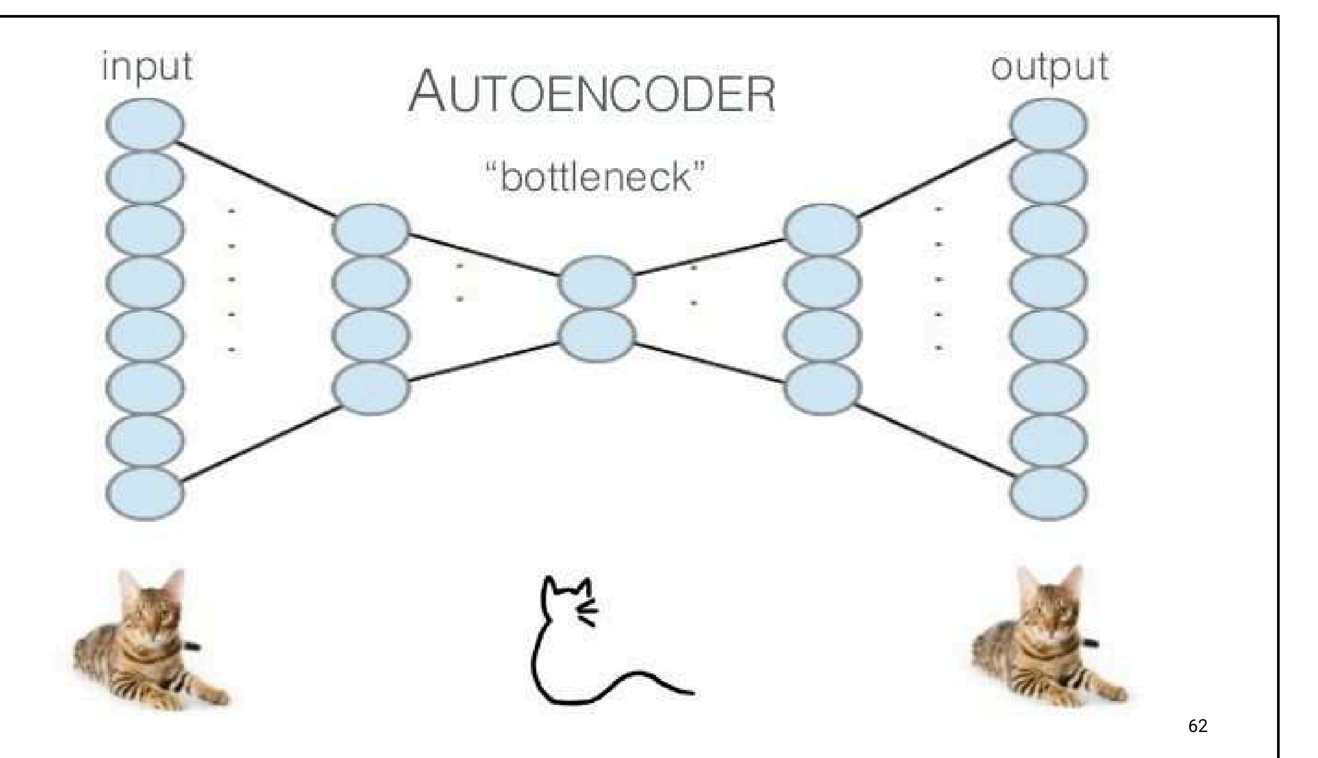
  📸 **Proceso de Compresión**:

  - **Imagen Original**:
    - Foto de 1000x1000 píxeles (1 millón de píxeles)
    - Cada píxel con información de color RGB
    - Ocupa mucho espacio de almacenamiento

  🔄 **Funcionamiento del Autoencoder**:

  1. **Encoder** (Compresión):
     - Reduce la imagen a 100x100 (10,000 píxeles)
     - Mantiene características esenciales:
       - 🎨 Colores principales
       - 📐 Formas básicas
       - 🖼️ Patrones importantes
  2. **Decoder** (Reconstrucción):
     - Intenta recrear la imagen original
     - Recupera la resolución 1000x1000
     - La imagen final puede perder algunos detalles finos, pero mantiene la esencia visual

[61] Imágen de https://stable-diffusion-art.com/how-stable-diffusion-work/



62

Es similar a cuando enviamos una foto por WhatsApp: la app comprime la imagen para que ocupe menos espacio pero sigue siendo reconocible y útil, aunque haya perdido algo de calidad en el proceso.

Se utilizan extensivamente en varios campos: en sistemas de recomendación para comprimir información de usuarios y productos, en procesamiento de imágenes médicas para mejorar la calidad de radiografías y resonancias, en sistemas de detección de fraude para identificar transacciones anómalas, y en el preprocesamiento de datos para otros modelos de machine learning.

- **Transformers:** son redes neuronales basadas en mecanismos de atención que han revolucionado el procesamiento del lenguaje natural y otras áreas, con ejemplos notables como BERT y GPT. A diferencia de las redes neuronales tradicionales o recurrentes, los transformers pueden procesar toda la secuencia de entrada simultáneamente, lo que los hace más eficientes y efectivos para capturar relaciones de largo alcance en los datos. Ya hemos hablado de Transformers en secciones anteriores y le volveremos a dedicar una sección completa más adelante.

Ventajas del Deep Learning:

- **Manejo de datos complejos**: Los modelos de deep learning pueden procesar y encontrar patrones en datos altamente complejos y no estructurados, como imágenes de alta resolución, videos, señales de audio o texto en lenguaje natural. Esta capacidad supera significativamente a los métodos tradicionales de machine learning que suelen requerir datos más estructurados y preprocesados.

- **Aprendizaje automático de *features*/características**: A diferencia de los métodos tradicionales donde los expertos deben diseñar manualmente las *features*/características relevantes, las redes neuronales *deep learning* aprenden automáticamente las representaciones jerárquicas más

[62] Imágen de https://www.cs.us.es/~fsancho/Blog/posts/VAE.md

importantes de los datos. Cada capa de la red aprende características cada vez más abstractas y complejas, desde bordes simples hasta conceptos completos.



- **Escalabilidad eficiente**: El rendimiento de los modelos de *deep learning* generalmente mejora con más datos y computación, aprovechando efectivamente el hardware moderno como GPUs y TPUs. Esto permite entrenar modelos cada vez más grandes y sofisticados que pueden capturar patrones más complejos en los datos.

- **Versatilidad y transferibilidad**: Los modelos de *deep learning* pueden aplicarse a una amplia gama de problemas y dominios diferentes. Además, el conocimiento aprendido en una tarea puede transferirse a otras tareas relacionadas mediante técnicas como el transfer learning, reduciendo significativamente el tiempo y los recursos necesarios para nuevas aplicaciones.

Desafíos del Deep Learning:

- **Necesidad de datos etiquetados**: La mayoría de los modelos requieren grandes cantidades de datos etiquetados para su entrenamiento. Crear estos conjuntos de datos puede ser extremadamente costoso y consumir mucho tiempo, especialmente en dominios especializados como el médico donde se requiere experiencia específica para el etiquetado. Además, la calidad y representatividad de estos datos es crucial para el rendimiento del modelo.

- **Recursos computacionales intensivos**: El entrenamiento de modelos profundos requiere una infraestructura computacional significativa. Los costos incluyen no solo el hardware especializado (GPUs, TPUs), sino también el consumo de energía asociado y los recursos de almacenamiento. El tiempo de entrenamiento puede variar desde días hasta semanas para modelos más complejos, lo que aumenta los costos operativos.

- **Interpretabilidad limitada**: Los modelos de *deep learning* son notoriamente difíciles de interpretar debido a su naturaleza de "caja negra". Esto presenta desafíos significativos en aplicaciones críticas

como diagnóstico médico, sistemas judiciales o financieros, donde es crucial entender y justificar las decisiones del modelo. Las técnicas actuales de interpretabilidad todavía son limitadas y no siempre proporcionan explicaciones satisfactorias.



En el contexto del deep learning, el concepto de "caja negra" refleja la complejidad y el poder de los modelos avanzados, como las redes neuronales profundas, que pueden procesar datos de entrada y transformarlos en soluciones excepcionales a problemas complejos. Aunque los detalles internos del proceso pueden ser difíciles de interpretar, esto no es una limitación insalvable, sino una invitación a desarrollar herramientas y técnicas que mejoren la explicabilidad. A medida que avanzamos, el desafío de entender estas "cajas negras" impulsa la innovación y abre nuevas posibilidades para aplicar inteligencia artificial de manera confiable y ética en campos como la medicina, las finanzas y la educación.

- **Riesgo de sobreajuste**: Los modelos pueden memorizar los datos de entrenamiento en lugar de aprender patrones generalizables, especialmente cuando el conjunto de datos es pequeño o no representativo. Esto lleva a un rendimiento pobre en datos nuevos. Controlar el sobreajuste requiere técnicas específicas de regularización, validación cruzada y una cuidadosa arquitectura del modelo, lo que aumenta la complejidad del proceso de desarrollo.

Con esto finalizamos nuestra sección sobre los conceptos básicos de las redes neuronales y el *deep learning*.

Ahora que ya tenés una mejor idea tanto de machine learning (supervisado y no supervisado), *reinforcement learning* y *deep learning*, es momento de comenzar con temas más concretos de cómo estas técnicas resuelven problemas y avanzan el campo de la inteligencia artificial. Tal vez el más beneficiado de ellos y el que se encuentra pasando por su mayor auge en la actualidad sea el del procesamiento de lenguaje natural (NLP por Natural Language Processing). Veamos por qué.

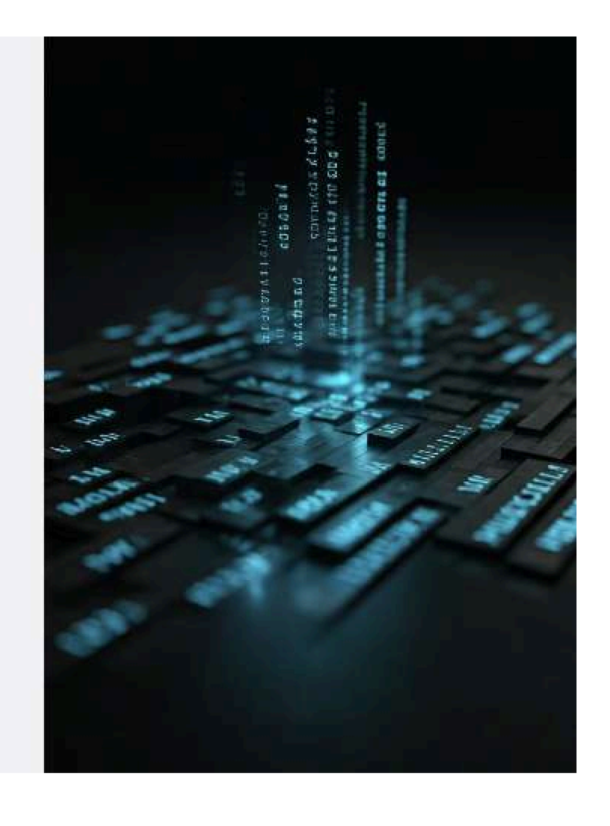
07. 🗣️ Procesamiento de Lenguaje Natural (NLP)

Posiblemente la disciplina dentro del campo de la Inteligencia Artificial que más se ha beneficiado por el progreso reciente, el procesamiento de lenguaje natural es uno de los desafíos más conocidos de la IA y uno que parece ser que ya hemos conseguido dominar a niveles insospechados hasta hace no muy poco. Capaz de revolucionar al mundo al punto en que hoy en día *conversar* con una inteligencia artificial nos parece algo natural, veamos de qué se trata exactamente.

Procesamiento de Lenguaje Natural (NLP)

El NLP permite a las máquinas entender y generar lenguaje humano. Es fundamental para aplicaciones como chatbots y traducción automática.

Facilita la interacción natural entre humanos y máquinas. Se basa en lingüística computacional y aprendizaje automático.



El objetivo del NLP es permitir que las máquinas comprendan, interpreten y generen lenguaje humano de manera *significativa*[63]. Esta tecnología es crucial para una amplia variedad de aplicaciones que van desde la clasificación y traducción automática de textos hasta los asistentes virtuales y el análisis de sentimientos.

El NLP se basa en una combinación de lingüística computacional y modelos de *machine learning* para procesar y analizar grandes cantidades de datos textuales. Esto incluye la comprensión de la estructura y el *significado* del lenguaje, así como la capacidad de generar respuestas coherentes y contextualmente apropiadas.

Lo primero que hay que entender cuando pensamos en NLP es que las *máquinas* no entienden palabras sino fundamentalmente matemáticas y operaciones numéricas porque los procesadores están diseñados para manipular números en su forma más básica.

---

[63] En este contexto, "significativa" se refiere a la capacidad de las máquinas para procesar y producir lenguaje de una manera que sea contextualmente apropiada, coherente y útil para los seres humanos. Esto implica no solo entender el significado literal de las palabras, sino también captar matices, intenciones y significados implícitos, así como generar respuestas que sean relevantes y adaptadas al contexto de la comunicación.



1. **El lenguaje binario:** En su nivel más básico, un procesador solo entiende dos estados: encendido (1) y apagado (0). Esto se debe a que los transistores, que son los componentes básicos de un procesador, sólo pueden estar en uno de estos dos estados.
2. **Representación binaria de números**
    - Los números se representan naturalmente en binario
    - Por ejemplo, el número 9 en binario es 1001
    - Esta representación es directa y eficiente para el procesador
3. **Arquitectura del procesador:** Los procesadores tienen circuitos especializados (ALU - Unidad Aritmético Lógica) diseñados específicamente para:
    - Sumar números
    - Restar números
    - Multiplicar números
    - Realizar operaciones lógicas (AND, OR, NOT)
4. **De números a todo lo demás:** Todo lo que vemos en una computadora debe traducirse a números:
    - Letras → números (ASCII/Unicode)
    - Colores → números (RGB)
    - Sonidos → números (frecuencias)
    - Imágenes → matrices de números
5. **¿Por qué no diseñar procesadores que entiendan directamente palabras?**
    - Sería extremadamente complejo y poco eficiente
    - Los números son universales y pueden representar cualquier tipo de información
    - Las operaciones matemáticas son la base para construir operaciones más complejas

Entonces al trabajar en NLP es como si tuviéramos una calculadora muy potente que solo sabe hacer operaciones matemáticas. Para que esta calculadora pueda "entender" palabras, necesitamos primero convertir esas palabras en números con los que pueda operar. Esto se logra mediante técnicas como la *vectorización* de palabras y la codificación de *tokens*, que transforman el texto en vectores numéricos. Estas representaciones capturan las relaciones semánticas y sintácticas entre las palabras, permitiendo a los

algoritmos de *machine learning* analizar y manipular el lenguaje de manera eficiente.

Componentes del NLP

Tokenización: La tokenización es el proceso de dividir el texto en unidades más pequeñas llamadas *tokens*, que pueden ser palabras, frases o incluso caracteres. Este es el primer paso en la mayoría de las tareas de NLP y es crucial para el análisis posterior.



Etiquetado de Partes del Discurso (POS Tagging): Este proceso asigna etiquetas gramaticales a cada token, como sustantivo, verbo, adjetivo, etc. Esto ayuda a entender la estructura gramatical del texto.

Ejemplo 1: Oración simple

"El gato negro duerme plácidamente."

El	gato	negro	duerme	plácidamente	.
DET	NOUN	ADJ	VERB	ADV	PUNCT

Ejemplo 2: Oración con diferentes tiempos verbales

"Los científicos han descubierto una nueva especie de planta en la selva amazónica."

Los	científicos	han	descubierto	una	nueva	especie	de	planta	en	la	selva	amazónica	.
DET	NOUN	AUX	VERB	DET	ADJ	NOUN	ADP	NOUN	ADP	DET	NOUN	ADJ	PUNCT



Lematización y Stemmatización: La *lematización* reduce las palabras a su *forma base* o *lema*, mientras que la *stemmatización* corta los sufijos para obtener la *raíz* de las palabras. Ambos procesos ayudan a normalizar las variaciones de palabras para un análisis más consistente.

Ejemplo 1: Verbos

Palabra Original	Lematización	Stemmatización
corriendo	correr	corr
corrió	correr	corr
correremos	correr	corr

Ejemplo 2: Sustantivos y Adjetivos

Palabra Original	Lematización	Stemmatización
mejores	bueno	mejor
niños	niño	niñ
ciudades	ciudad	ciudad

Ejemplo 3: Palabras Irregulares

Palabra Original	Lematización	Stemmatización
fue	ser/ir	fu
soy	ser	soy
mejor	bueno	mejor

Reconocimiento de Entidades Nombradas (NER): NER identifica y clasifica *entidades* nombradas en el texto, como nombres de personas, lugares, organizaciones, fechas, etc. Esto es útil para extraer información estructurada del texto no estructurado.

Ejemplo 1: Noticia

Elon Musk, CEO de **Tesla** y **SpaceX**, anunció el **15 de marzo de 2023** que la compañía planea construir una nueva fábrica en **Austin, Texas**. El anuncio se realizó durante el **Tesla Investor Day**.

Ejemplo 2: Reseña de producto

El nuevo **iPhone 13 Pro** de **Apple** fue lanzado en **septiembre de 2021**. Con su potente chip **A15 Bionic** y una impresionante cámara triple, este dispositivo ha recibido excelentes críticas en **Estados Unidos** y **Europa**.

Ejemplo 3: Evento histórico

La **Revolución Francesa** comenzó el **14 de julio de 1789** con la toma de la **Bastilla** en **París**. **Napoleón Bonaparte**, quien más tarde se convertiría en emperador de **Francia**, jugó un papel crucial en los eventos posteriores a la revolución.

Persona Organización Ubicación Fecha Evento Producto

Análisis Sintáctico (Parsing): El análisis sintáctico determina la estructura gramatical del texto, creando un árbol sintáctico que representa la relación entre las palabras y las frases.

Oración: "Los estudiantes aplicados que asisten a la universidad regularmente obtienen mejores calificaciones en sus exámenes finales."



Análisis de la estructura:

Sujeto: "Los estudiantes aplicados que asisten a la universidad regularmente"

- "Los": Artículo que determina el sustantivo
- "estudiantes": Sustantivo núcleo del sujeto
- "aplicados": Adjetivo que modifica a "estudiantes"
- "que asisten a la universidad regularmente": Oración subordinada adjetiva que modifica a "estudiantes"

Predicado: "obtienen mejores calificaciones en sus exámenes finales"

- "obtienen": Verbo principal
- "mejores calificaciones": Objeto directo
- "en sus exámenes finales": Complemento circunstancial de lugar

Análisis Semántico: Este componente se centra en la comprensión del significado del texto, incluyendo la resolución de ambigüedades y la inferencia de información implícita.

Frase a analizar: "Juan vio al hombre con el telescopio"

Interpretaciones posibles:

Interpretación 1:

Juan usó un telescopio para ver al hombre.

- Sujeto: Juan
- Acción: ver
- Objeto: al hombre
- Instrumento: con el telescopio

Interpretación 2:

Juan vio a un hombre que tenía un telescopio.

- Sujeto: Juan
- Acción: ver
- Objeto: al hombre con el telescopio (el hombre tiene el telescopio)

Resolución de ambigüedad:

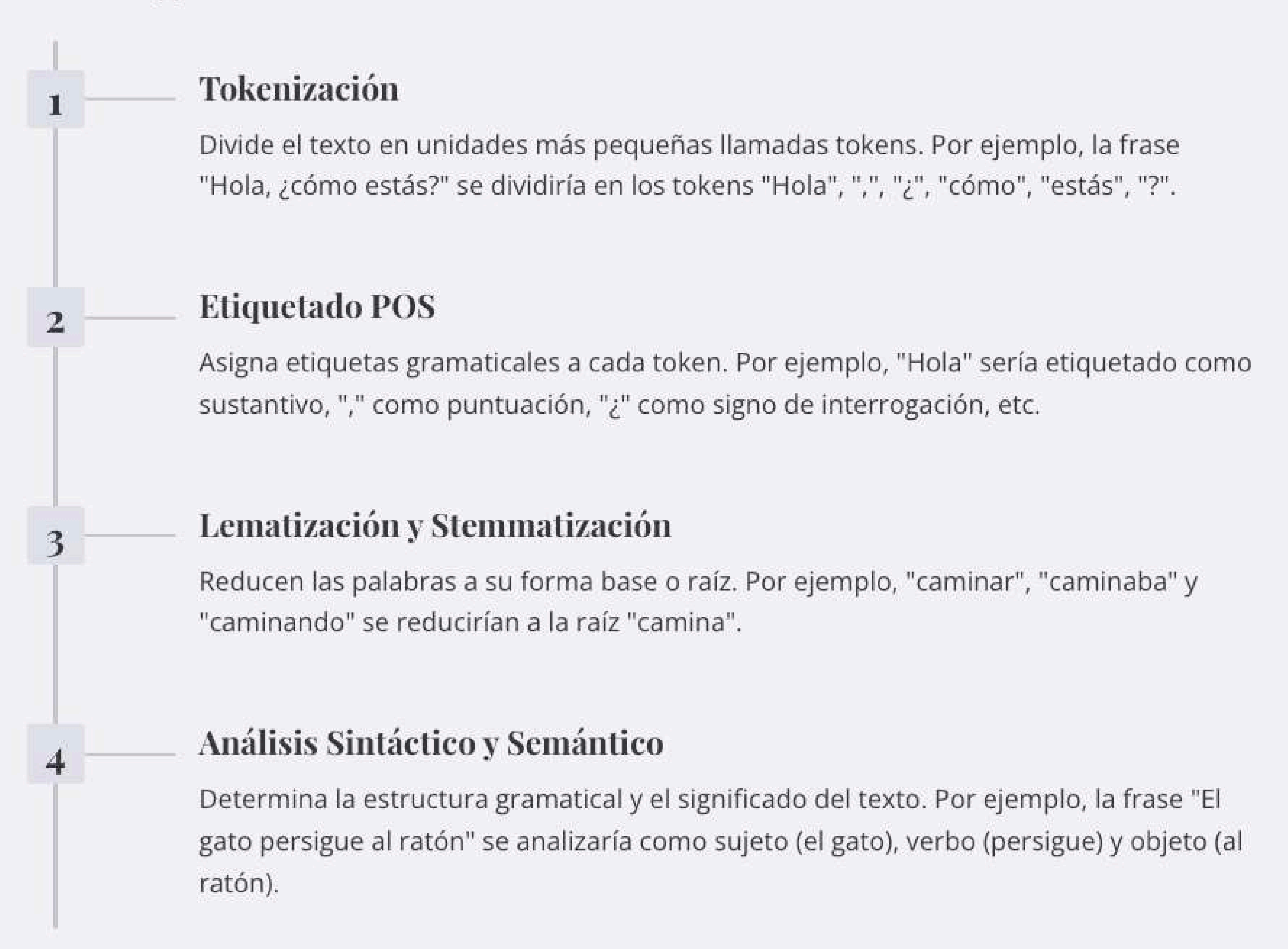
Para resolver esta ambigüedad, el sistema de NLP necesitaría contexto adicional. Por ejemplo:

- "Juan estaba en el observatorio" sugeriría la Interpretación 1.
- "Juan estaba en la playa" podría sugerir la Interpretación 2.

Inferencias posibles:

- Juan tiene buena vista (capaz de ver a distancia).
- El hombre estaba lejos (necesidad de un telescopio).
- La escena ocurre durante el día o en un lugar iluminado (capacidad de ver).

Repasemos:



Una vez que entendemos los componentes fundamentales del NLP como la tokenización, el etiquetado POS, la lematización, el NER y el análisis sintáctico y semántico, podemos apreciar cómo cada uno de estos elementos contribuye de manera única al procesamiento del lenguaje natural.

- La tokenización sirve como piedra angular, transformando el texto en unidades manejables que permiten un análisis granular del lenguaje, facilitando tareas como la búsqueda de palabras clave y el análisis de frecuencias.
- El etiquetado de partes del discurso (POS) añade una capa crucial de información gramatical que permite entender el rol de cada palabra en su contexto, lo que es esencial para tareas como la traducción automática y la generación de texto natural.
- La lematización y stemmatización, aunque aparentemente simples, son fundamentales para reducir la complejidad del lenguaje al normalizar las variantes de palabras, mejorando significativamente la eficacia de los sistemas de búsqueda y clasificación de textos.
- El reconocimiento de entidades nombradas (NER) va más allá del análisis superficial, permitiendo identificar y clasificar elementos críticos como nombres, lugares y organizaciones, lo que es vital para la extracción de información y la comprensión contextual.
- El análisis sintáctico proporciona la estructura que conecta estos elementos, revelando las relaciones gramaticales que dan coherencia al texto, mientras que el análisis semántico corona

estos procesos al interpretar el significado real de las palabras y frases en su contexto.

Estos componentes, trabajando en conjunto, crean un sistema robusto que permite a las máquinas procesar y comprender el lenguaje humano de manera cada vez más sofisticada, estableciendo las bases para aplicaciones avanzadas de NLP. Sin embargo, para hacer estos componentes verdaderamente útiles, necesitamos traducir el texto procesado en un formato que las máquinas puedan entender y manipular. Este desafío fundamental llevó al desarrollo de diferentes técnicas de representación numérica del texto, cada vez más sofisticadas, que permiten convertir las palabras y sus relaciones en vectores numéricos. Esta transformación es crucial porque, como vimos anteriormente, las computadoras sólo pueden procesar números en su forma más básica. La evolución de estas técnicas de representación ha sido fundamental para mejorar la capacidad de las máquinas para procesar el lenguaje natural de manera más efectiva, permitiendo capturar gradualmente más matices y complejidades del lenguaje humano. Veamos las principales técnicas y modelos que han marcado este progreso, comenzando con los métodos más básicos hasta llegar a los avances más recientes que están revolucionando el campo.

Técnicas de vectorización de texto

One-Hot Encoding: Es la forma más básica de convertir palabras en números. Consiste en representar cada palabra como un vector donde todos los elementos son 0 excepto uno que es 1, indicando la posición de esa palabra en nuestro vocabulario.

Por ejemplo, si tenemos el vocabulario ["gato", "perro", "casa", "el", "come", "duerme"], cada palabra se representaría así:

- "gato" → [1, 0, 0, 0, 0, 0]
- "perro" → [0, 1, 0, 0, 0, 0]
- "casa" → [0, 0, 1, 0, 0, 0]
- "el" → [0, 0, 0, 1, 0, 0]
- "come" → [0, 0, 0, 0, 1, 0]
- "duerme" → [0, 0, 0, 0, 0, 1]

Si queremos representar la oración "el gato come", cada palabra se convierte en su vector correspondiente:

- "el" → [0, 0, 0, 1, 0, 0]
- "gato" → [1, 0, 0, 0, 0, 0]
- "come" → [0, 0, 0, 0, 1, 0]

La oración completa se podría representar como una matriz:

[0, 0, 0, 1, 0, 0] # el

[1, 0, 0, 0, 0, 0] # gato

[0, 0, 0, 0, 1, 0] # come

Las principales limitaciones de esta técnica son que:

- No captura ninguna relación semántica entre palabras (para el modelo, todas las palabras son igualmente diferentes). Por ejemplo, "gato" y "perro" son tan diferentes entre sí como "gato" y "logaritmo" según esta representación, cuando intuitivamente sabemos que "gato" y "perro" están más relacionados al ser ambos animales domésticos. Esta limitación hace que el modelo no pueda aprovechar similitudes naturales entre palabras relacionadas.
- Genera vectores muy grandes y dispersos (con muchos ceros). Imaginemos un diccionario de 10,000

palabras - cada palabra se representaría con un vector de 10,000 números donde 9,999 son ceros y solo uno es 1. Esto es computacionalmente ineficiente, como si tuviéramos una biblioteca enorme con miles de estantes pero solo usáramos uno para cada libro, desperdiciando todo el espacio restante.

- No es eficiente para vocabularios grandes - imagina un vocabulario de 50,000 palabras, ¡cada palabra sería un vector de 50,000 dimensiones! Para ponerlo en perspectiva, si quisiéramos procesar un texto de 100 palabras, terminaríamos manejando una matriz de 100 x 50,000 números, donde la gran mayoría serían ceros. Es como si para identificar cada persona en una ciudad necesitáramos una lista con todos los habitantes donde marcamos "sí" solo para esa persona y "no" para todos los demás - un sistema extremadamente ineficiente.

A pesar de sus limitaciones, One-Hot Encoding es fundamental para entender cómo convertimos palabras en números y sirve como base para entender técnicas más avanzadas como BoW, TF-IDF y Word Embeddings.

Bag of Words (BoW): Mientras que One-Hot Encoding representa cada palabra individual como un vector de ceros y un uno, Bag of Words (BoW) da un paso más allá y representa documentos completos contando la frecuencia de cada palabra. Esta técnica es más eficiente que One-Hot Encoding porque:

1. En lugar de necesitar múltiples vectores para representar un texto (uno por cada palabra), BoW utiliza un único vector que resume todo el documento.
2. Nos permite comparar documentos completos de manera directa, independientemente de su longitud.
3. Captura información sobre la importancia relativa de las palabras basada en su frecuencia de aparición.

Por ejemplo, si tenemos el texto "el gato come y el gato duerme", BoW lo representaría como un único vector:

- "el": 2 veces
- "gato": 2 veces
- "come": 1 vez
- "y": 1 vez
- "duerme": 1 vez

Sin embargo, esta técnica todavía tiene limitaciones importantes. Consideremos estas dos oraciones:

- "El gato persigue al ratón"
- "El ratón persigue al gato"

Para un humano, estas oraciones tienen significados muy diferentes. Sin embargo, para BoW serían exactamente iguales, porque contienen las mismas palabras con la misma frecuencia:

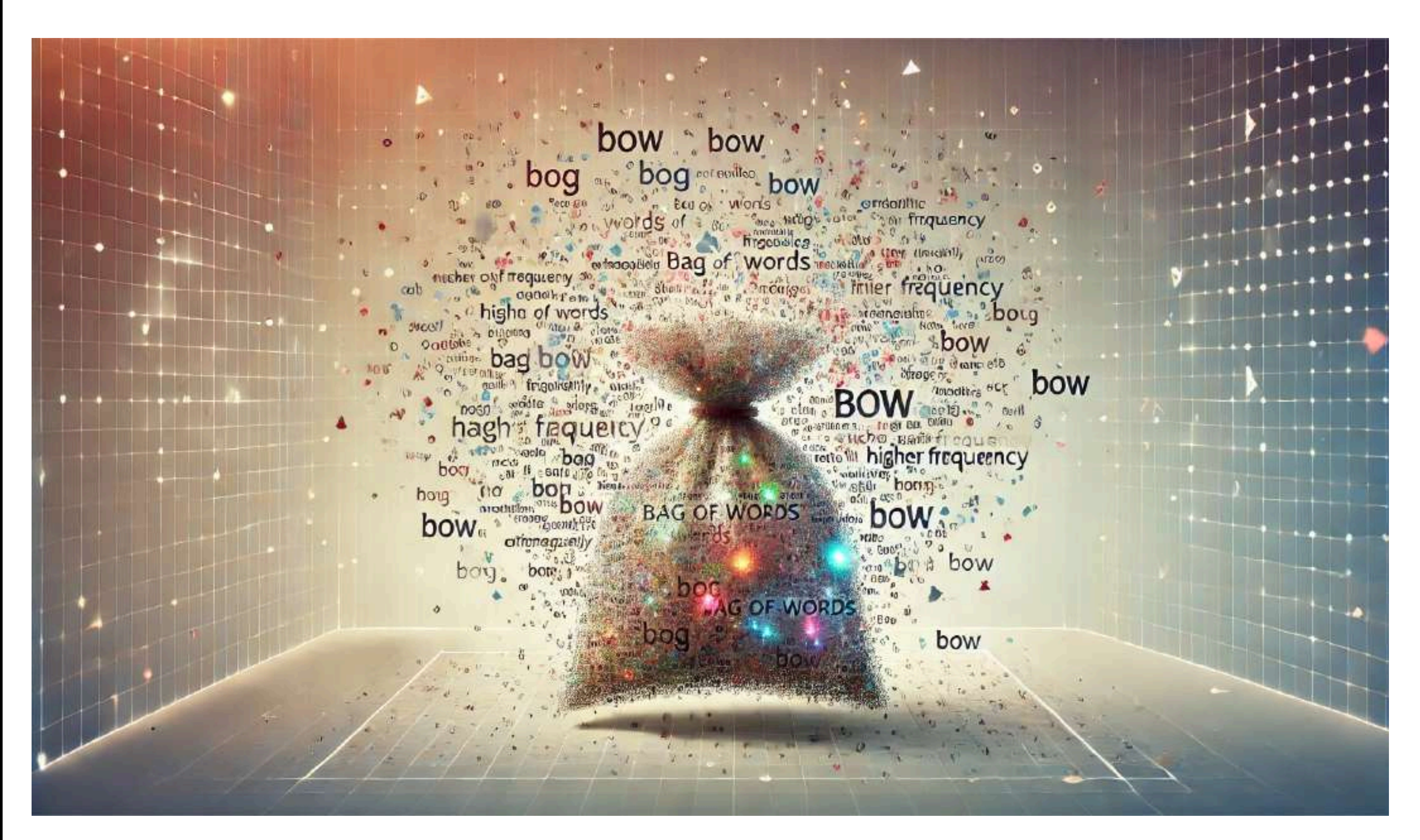
- "el": 1 vez
- "gato": 1 vez
- "persigue": 1 vez
- "al": 1 vez
- "ratón": 1 vez

Esta es la principal limitación de BoW: pierde información crucial sobre el significado y la estructura del texto. Es como si tuviéramos los ingredientes de una receta, pero sin las instrucciones de cómo combinarlos.

A pesar de esta limitación, BoW es útil para tareas básicas como:

- Clasificación de documentos por tema: Esto funciona porque los documentos de un mismo tema tienden a compartir un vocabulario común. Por ejemplo, los artículos de deportes usarán frecuentemente palabras como "gol", "partido", "jugador", mientras que los de cocina tendrán más menciones de "receta", "ingredientes", "hornear" - incluso si no captamos el orden exacto de las palabras, la frecuencia de este vocabulario específico es un buen indicador del tema.
- Filtrado de spam: Los correos spam suelen usar ciertas palabras clave de manera repetitiva ("oferta", "gratis", "gana dinero", "urgente"). No importa tanto el orden en que aparezcan estas palabras, su mera presencia y frecuencia ya es una señal fuerte de que el correo podría ser spam.
- Análisis básico de sentimientos: Las palabras positivas y negativas tienden a dominar el tono de un texto independientemente de su orden. Por ejemplo, un texto con alta frecuencia de palabras como "excelente", "maravilloso", "me encanta" probablemente exprese un sentimiento positivo, mientras que uno con muchas apariciones de "terrible", "pésimo", "odio" será negativo.
- Búsqueda de documentos similares: Si dos documentos tratan sobre el mismo tema, probablemente compartirán un vocabulario similar. Por ejemplo, dos reseñas de un mismo restaurante probablemente mencionarán palabras similares sobre la comida, el servicio y el ambiente, aunque las frases estén estructuradas de manera diferente.

Su simplicidad lo hace fácil de implementar y entender, lo que lo convierte en un buen punto de partida para introducirse en el procesamiento del lenguaje natural, aunque para tareas más complejas se necesiten técnicas más avanzadas.



Ejemplo de BoW

Frases de ejemplo:

1. "El gato come pescado"
2. "El perro come carne"
3. "El gato y el perro comen"

Representación BoW:

Palabra	Frase 1	Frase 2	Frase 3
el	1	1	2
gato	1	0	1
come	1	1	0
pescado	1	0	0
perro	0	1	1
carne	0	1	0
y	0	0	1
comen	0	0	1

Ventajas de BoW:

- Simple y fácil de implementar
- Eficiente para grandes volúmenes de texto
- Útil para clasificación de textos
- Base para técnicas más avanzadas

Limitaciones de BoW:

- Pierde el orden de las palabras
- No capta el contexto o significado
- No maneja bien palabras nuevas
- Puede generar vectores muy grandes

TF-IDF (Term Frequency-Inverse Document Frequency): TF-IDF es como una versión más inteligente del Bag of Words. En lugar de simplemente contar palabras, TF-IDF evalúa qué tan importante es una palabra para un documento dentro de una colección de documentos.

Imaginemos que tenemos tres reseñas de restaurantes:

1. "La pasta estaba riquísima. El restaurante tiene buena pasta."

2. "La hamburguesa estaba riquísima. El restaurante es acogedor."
3. "El restaurante tiene buena ubicación."

TF-IDF funciona en dos partes:

1. **TF (Term Frequency)** - ¿Cuántas veces aparece la palabra?
   - En la reseña 1, "pasta" aparece 2 veces
   - "restaurante" aparece 1 vez en cada reseña
2. **IDF (Inverse Document Frequency)** - ¿Qué tan única es la palabra en todos los documentos?
   - "pasta" solo aparece en 1 de 3 reseñas → **más importante**, porque es un término discriminativo. Es como cuando describes a alguien: decir que tiene "pelo" no ayuda mucho a identificarlo porque casi todos tienen pelo, pero mencionar que tiene "rastas azules" es mucho más útil porque es una característica distintiva. De la misma manera, "pasta" nos ayuda a distinguir específicamente el primer restaurante de los demás, nos dice algo único sobre él.
   - "restaurante" aparece en las 3 reseñas → **menos importante**, porque es un término demasiado común que no aporta información distintiva. Es como decir que una persona "tiene ojos" - es verdad, pero no nos ayuda a distinguirla de otras personas porque todos tienen ojos. De manera similar, decir que un lugar es un "restaurante" en una colección de reseñas de restaurantes no nos ayuda a diferenciarlo de los demás, no nos dice nada específico sobre ese lugar en particular.

Así, TF-IDF daría:

- Mayor importancia a "pasta" en la reseña 1 porque:
  - Aparece frecuentemente (TF alto)
  - Es única a esa reseña (IDF alto)
- Menor importancia a "restaurante" porque:
  - Aparece en todas las reseñas (IDF bajo)

Esta técnica es especialmente útil para:

- Búsquedas en internet: Al darle más peso a palabras únicas y específicas, TF-IDF ayuda a encontrar los resultados más relevantes. Por ejemplo, si buscas "receta pasta carbonara", los documentos que mencionan "carbonara" (una palabra específica que aparece en pocos documentos) tendrán prioridad sobre aquellos que solo mencionan "pasta" o "receta" (palabras más comunes). Es como si al buscar un libro en una biblioteca, le diéramos más importancia a las palabras únicas del título que a palabras comunes como "el" o "de".
- Recomendación de contenido: Si leíste un artículo sobre "cocina italiana", TF-IDF puede identificar otros artículos que comparten términos específicos importantes como "risotto" o "pesto", en lugar de recomendar cualquier artículo que simplemente mencione "comida" o "cocina" muchas veces. Es como si un amigo te recomendara una película no solo porque es de "acción", sino porque comparte elementos distintivos específicos con las películas que te gustan, como "espionaje internacional" o "persecuciones en ciudades europeas".
- Resumen de textos: TF-IDF puede identificar las oraciones que contienen las palabras más importantes y distintivas del texto, creando resúmenes que capturan la esencia única del documento. Es como cuando describís una película: probablemente vas a mencionar los elementos únicos y memorables, no las partes comunes que podrían aplicar a cualquier película.
- Identificación de palabras clave en documentos: Al identificar qué términos son realmente distintivos en un documento, TF-IDF puede extraer palabras clave que verdaderamente representan su contenido único. Por ejemplo, en un artículo sobre "El Rey León", términos específicos como "Simba", "Mufasa" o "savana africana" serán identificados como más importantes que palabras comunes como "película" o "historia". Es como cuando describís a un amigo una película que viste:

naturalmente destacas los elementos únicos que la distinguen (como los personajes principales o el escenario específico), en lugar de mencionar aspectos generales que podrían aplicar a cualquier película.

La principal ventaja de TF-IDF sobre BoW es que puede identificar las palabras que realmente caracterizan a cada documento, ignorando las palabras comunes que no aportan información distintiva.

Ejemplo de TF-IDF

Documentos de ejemplo:

1. El gato come pescado fresco
2. El perro come carne fresca
3. El gato y el perro juegan en el parque

Fórmulas:

TF(t,d) = (Número de veces que t aparece en d) / (Número total de términos en d)

IDF(t) = log(Número total de documentos / Número de documentos que contienen t)

TF-IDF(t,d) = TF(t,d) * IDF(t)

Cálculo de TF-IDF:

Término	TF (Doc1)	TF (Doc2)	TF (Doc3)	IDF	TF-IDF (Doc1)	TF-IDF (Doc2)	TF-IDF (Doc3)
gato	0.2	0	0.125	0.176	0.035	0	0.022
pescado	0.2	0	0	0.477	0.095	0	0
perro	0	0.2	0.125	0.176	0	0.035	0.022
fresco/fresca	0.2	0.2	0	0.176	0.035	0.035	0
parque	0	0	0.125	0.477	0	0	0.060

Interpretación:

- Términos como "pescado" y "parque" tienen valores TF-IDF más altos porque son más específicos de ciertos documentos.
- Palabras comunes como "el" o "y" (no mostradas) tendrían valores TF-IDF muy bajos debido a su alta frecuencia en todos los documentos.
- TF-IDF ayuda a identificar términos que son importantes para un documento específico dentro de una colección.

Ventajas de TF-IDF sobre BoW:

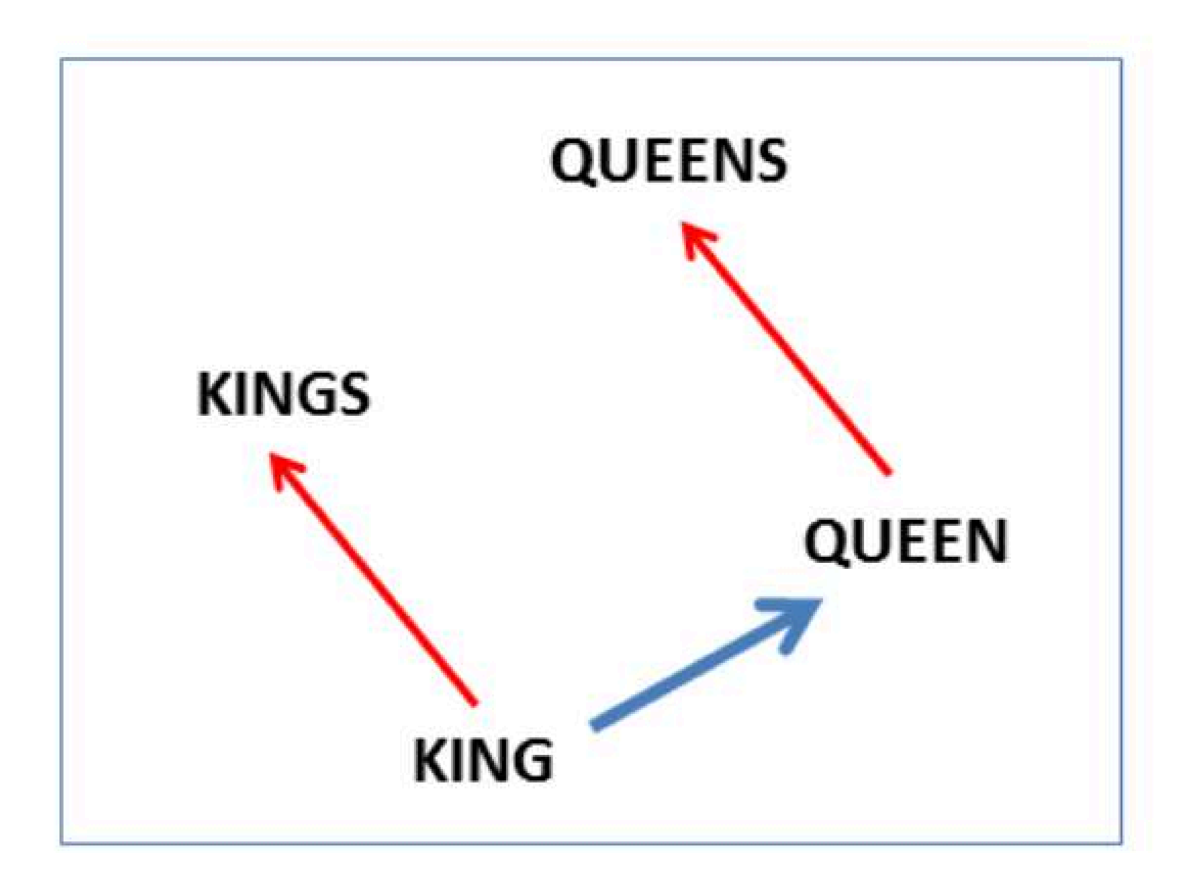
- Considera la importancia relativa de las palabras en el contexto de todos los documentos.
- Reduce la importancia de palabras muy comunes que aparecen en muchos documentos.
- Destaca términos que son más característicos de cada documento.
- Mejora la precisión en tareas como búsqueda de información y clasificación de textos.

Sin embargo, tanto BoW como TF-IDF siguen teniendo una limitación fundamental: no pueden capturar las relaciones semánticas entre las palabras. Como ya dijimos antes, para estas técnicas, palabras como "perro" y "gato" son tan diferentes entre sí como "perro" y "smartphone", cuando intuitivamente sabemos que las primeras están más relacionadas al ser ambos animales. Para abordar esta limitación, surgieron técnicas más avanzadas como los word embeddings, que revolucionaron la forma en que representamos las palabras al permitir capturar estas relaciones semánticas de manera matemática.

Word Embeddings: ¿Te acordás que en el comienzo del libro te comenté que uno de mis temas favoritos del mundo de la IA eran los Word Embeddings? Ahora nos toca analizarlos un poco más y los vas a poder apreciar aún más habiendo visto cómo los métodos anteriores (BoW y TF-IDF), para representar lenguaje de forma numérica, parecían tener muchas ineficiencias.

Los Word Embeddings, como ya vimos, son representaciones numéricas de palabras que capturan relaciones semánticas en un *espacio vectorial* continuo. Imaginá que pudieras ubicar cada palabra en un mapa multidimensional, donde la posición de cada palabra está determinada por su significado. En este mapa, palabras como "gato" y "perro" estarían cerca porque comparten muchas características (son mascotas, mamíferos, tienen cuatro patas), mientras que "gato" y "smartphone" estarían muy lejos porque tienen poco en común. A diferencia de codificaciones más simples como one-hot encoding, donde cada palabra es un vector de ceros y un único uno, los embeddings representan cada palabra como un vector denso (por ejemplo, [0.2, 0.5, -0.3]) donde cada número contribuye al significado. Es como si cada número en ese vector representara una característica o aspecto diferente de la palabra, y el conjunto de todos esos números nos da una "huella digital" única del significado de esa palabra.

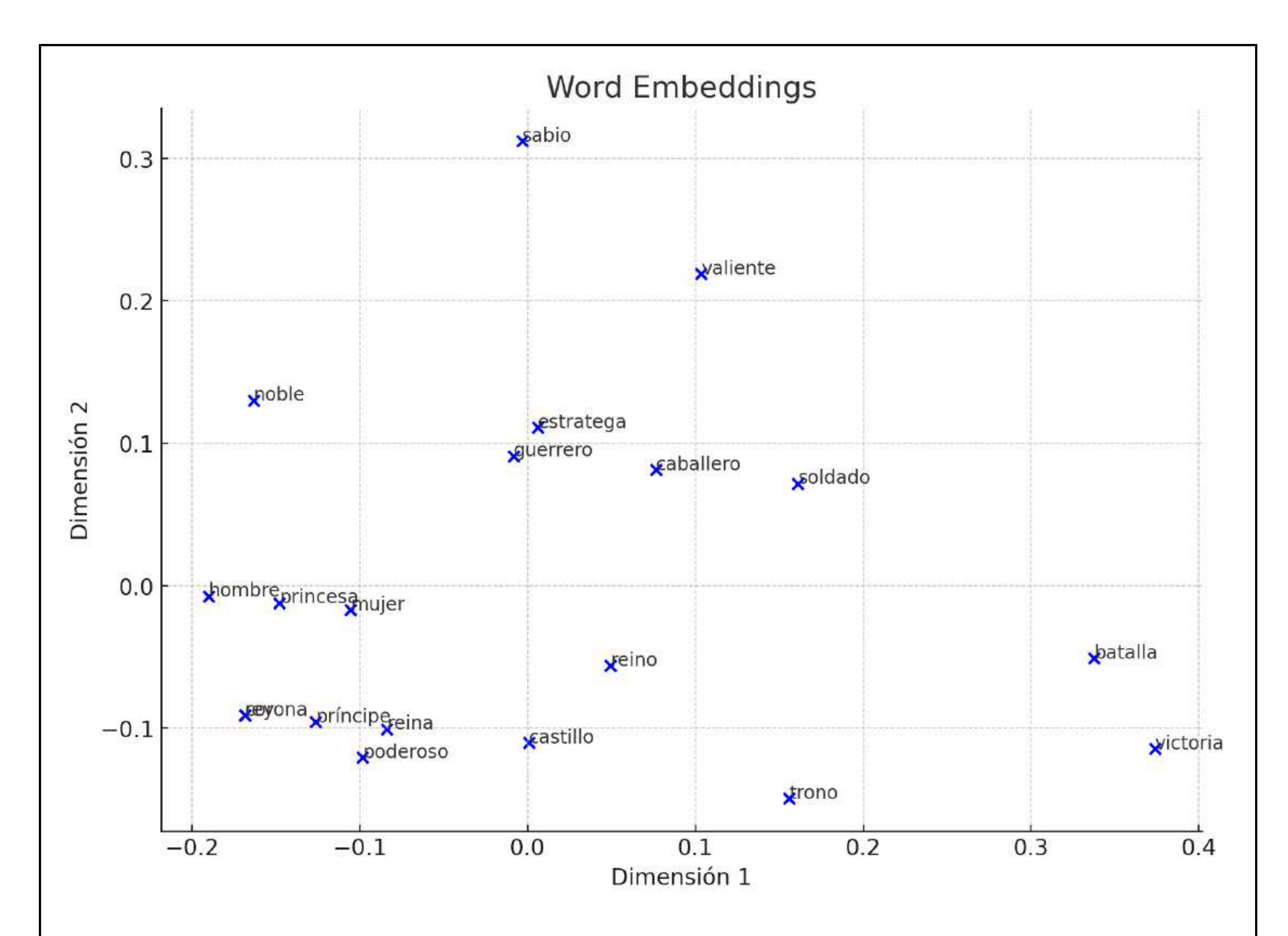
Lo fascinante de los Word Embeddings es que pueden capturar relaciones analógicas mediante operaciones matemáticas simples. Por ejemplo, la operación vectorial "rey - hombre + mujer ≈ reina" funciona porque el modelo aprende las relaciones semánticas entre palabras a partir de cómo aparecen en contextos similares.



Modelos como Word2Vec (desarrollado por Google) y GloVe (creado por Stanford) implementan estas ideas, permitiendo que las máquinas "entiendan" relaciones semánticas complejas a través de estas representaciones numéricas.

Características clave de Word Embeddings:

- **Representación *densa***: A diferencia de One-Hot Encoding y BoW que usan vectores dispersos (mayormente ceros), cada palabra se representa como un vector de números reales donde todos los valores son significativos. Por ejemplo, una palabra podría ser [0.2, -0.5, 0.8], donde cada número contribuye al significado.
- **Dimensionalidad reducida**: Mientras que BoW necesita un vector del tamaño del vocabulario (que puede ser de 50,000 o más dimensiones), los embeddings típicamente usan solo 100-300 dimensiones. Esta reducción dramática hace que el procesamiento sea mucho más eficiente sin perder información semántica importante.
- **Captura relaciones semánticas**: La magia de los embeddings está en que palabras con significados similares terminan cerca en el espacio vectorial. Por ejemplo, "rey" y "reina" tendrán vectores similares, y se pueden realizar operaciones como "rey - hombre + mujer ≈ reina", capturando relaciones analógicas.
- **Aprendizaje no supervisado**: Los embeddings se aprenden automáticamente al exponer el modelo a grandes cantidades de texto sin necesidad de etiquetas manuales. El modelo aprende las relaciones entre palabras observando cómo aparecen en contextos similares: si "perro" y "gato" aparecen en contextos parecidos (por ejemplo, "el ___ come", "el ___ duerme"), sus vectores serán similares.



En esta visualización de los Word Embeddings, las palabras relacionadas semánticamente, como "principe", "reina", "castillo", y "reino", están más cerca unas de otras en el espacio vectorial, lo que refleja su similitud en significado o contexto. A través de la reducción dimensional a 2D, podemos observar cómo conceptos similares tienden a agruparse, mientras que palabras menos relacionadas, como "batalla" o "sabio", se encuentran más alejadas.

Esto es útil en tareas de procesamiento de lenguaje natural (NLP) como traducción automática, análisis de sentimientos, clasificación de texto, y generación de texto. Al representar las palabras en un espacio vectorial, los modelos pueden identificar similitudes y diferencias entre ellas, permitiendo que el sistema "entienda" el contexto de las palabras y cómo se relacionan entre sí, mejorando la precisión y efectividad en varias aplicaciones de inteligencia artificial.

Los Word Embeddings representan un avance significativo en NLP, permitiendo una representación más rica y semánticamente significativa de las palabras. A diferencia de BoW y TF-IDF, que se centran en la frecuencia de las palabras, los embeddings capturan el significado y las relaciones entre palabras, lo que los hace muy valiosos en una amplia gama de aplicaciones de procesamiento de lenguaje natural.

Bag of Words (BoW)	**TF-IDF**	**Word Embeddings**
Representa frecuencias de palabras	Pondera importancia de palabras	Vector denso de baja dimensión
Vector disperso y de alta dimensión	Vector disperso	Captura relaciones semánticas
No captura significado	Captura relevancia en documentos	Permite operaciones algebraicas

Esta evolución en la representación de palabras, desde simples vectores binarios hasta embeddings capaces de capturar relaciones semánticas complejas, marcó un antes y un después en el procesamiento del lenguaje natural. Sin embargo, los word embeddings tradicionales tenían una limitación importante: cada palabra tenía siempre el mismo vector, independientemente del contexto en que apareciera. Por ejemplo, la palabra "banco" tendría la misma representación tanto en "me senté en el banco del parque" como en "fui al banco a depositar dinero". Esta limitación llevó al desarrollo de arquitecturas más sofisticadas como los transformers, que revolucionaron el campo al introducir embeddings contextuales - representaciones dinámicas que cambian según el contexto en que aparece cada palabra. Veamos cómo funcionan estas arquitecturas y por qué representaron un salto cualitativo en nuestra capacidad para procesar el lenguaje natural.

Transformers: Son las arquitecturas más avanzadas en NLP actual, popularizadas por modelos como BERT (de Google) y GPT (de OpenAI). Si los embeddings tradicionales son como tomar una fotografía fija de cada palabra, los Transformers son como una cámara de video que captura cómo cada palabra interactúa con todas las demás en tiempo real.

El corazón de estos modelos es el mecanismo de *atención*, que funciona como un sistema de "focos" que iluminan las conexiones importantes entre palabras, sin importar qué tan lejos estén en el texto. Por ejemplo, en la frase "El perro persiguió la pelota pero no la atrapó porque era muy rápida", el modelo puede conectar directamente "rápida" con "pelota" aunque estén separadas por varias palabras.

Los Transformers superan limitaciones cruciales de modelos anteriores:

- Pueden procesar todas las palabras de una frase en paralelo, en lugar de una por una
- Manejan exitosamente relaciones a larga distancia en el texto
- Generan respuestas más coherentes y contextualizadas

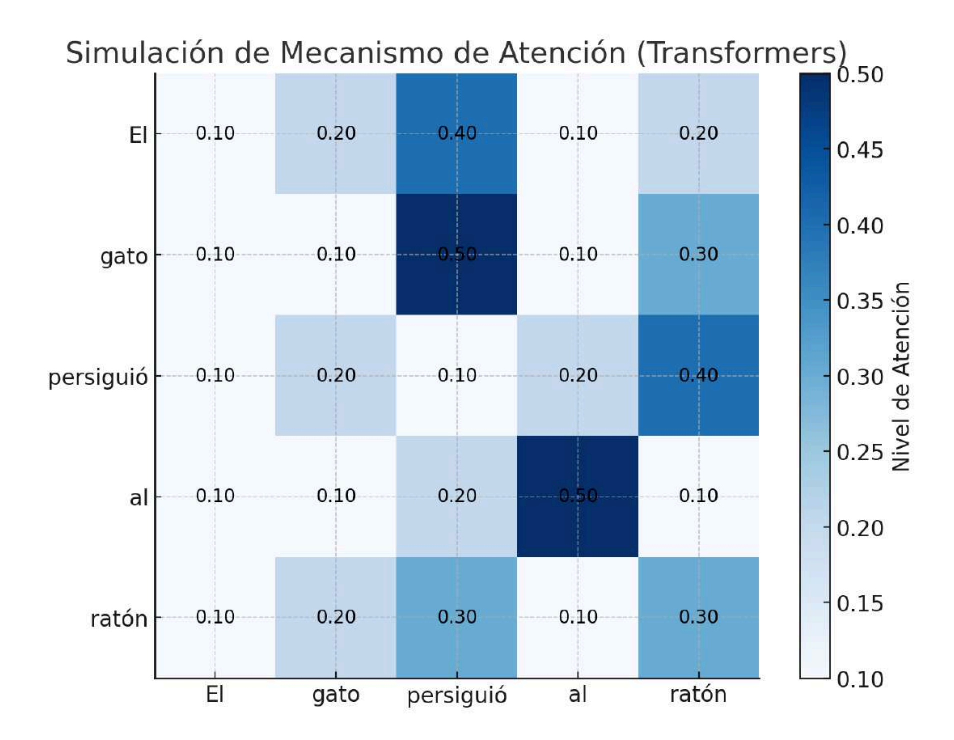
Esta arquitectura ha permitido avances revolucionarios como:

- Traducción automática de alta calidad
- Respuesta a preguntas en lenguaje natural
- Generación de texto que parece escrito por humanos
- Comprensión profunda del contexto y la intención en el lenguaje

Mecanismo de Atención

El mecanismo de atención permite al modelo enfocarse en diferentes partes del texto de entrada al procesar cada palabra. Esto es crucial para capturar relaciones contextuales a largo plazo.

- **Auto-atención:** Relaciona diferentes posiciones de una sola secuencia.
- **Atención cruzada:** Relaciona posiciones entre dos secuencias diferentes.



El gráfico muestra cómo cada palabra en la oración "El gato persiguió al ratón" pone atención en las demás. Por ejemplo, "gato" pone más atención en "persiguió" y "ratón", que son palabras claves en el contexto de la acción.

Los Transformers utilizan estos mecanismos de atención para entender cómo cada palabra en una oración se relaciona con otras, sin importar su posición. Esto permite que el modelo capte mejor el contexto y el significado de las palabras. Gracias a esto, los Transformers son muy útiles en tareas como la traducción automática, el análisis de texto y la generación de respuestas, ya que pueden enfocarse en las palabras más importantes y entender mejor el lenguaje.

El mecanismo de atención es el núcleo de los Transformers, que son la base de los Modelos de Lenguaje de Gran Escala (LLM) como GPT, Claude, Gemini, Llama, etc.

Características Clave de los Transformers:

- Procesamiento paralelo: A diferencia de las RNNs, procesa toda la secuencia a la vez.
- Captura de dependencias a largo plazo: Supera las limitaciones de distancia en RNNs.
- Aprendizaje contextual bidireccional: Considera el contexto completo (anterior y posterior).

- Escalabilidad: Puede entrenarse con enormes cantidades de datos.

Por esto y mucho más, tengo una sección entera dedicada a los Transformers un poco más adelante.

Aplicaciones del Procesamiento de Lenguaje Natural (NLP)

Traducción Automática
Traduce eficientemente textos entre múltiples idiomas con precisión y fluidez.

Asistentes Virtuales
Entienden y responden a comandos de voz de manera natural e intuitiva.

Análisis de Sentimientos
Identifica con precisión la opinión y el tono emocional en textos como reseñas y comentarios.

Modelos de Lenguaje Avanzados
Generan texto realista y comprenden el lenguaje natural en una amplia gama de aplicaciones empresariales y personales.

Como podés ver, el mundo Procesamiento de Lenguaje Natural es muy extenso y yo acá pretendo darte solo los conocimientos básicos, ¡espero haberlo conseguido! Además el NLP evolucionó muchísimo, desde los primeros modelos basados en reglas hasta las sofisticadas arquitecturas de Transformers que dominan el campo actualmente. Esta progresión, que abarca las técnicas que vimos como Bag of Words, TF-IDF, Word Embeddings y finalmente los Transformers, refleja nuestra creciente capacidad para hacer que las máquinas entiendan cada vez más el *contexto* y los detalles del lenguaje humano. Los avances en NLP revolucionaron cómo interactuamos con la tecnología, permitiendo las aplicaciones que vemos hoy en día.

Sin embargo, el lenguaje es sólo una faceta de la inteligencia humana. Así como hemos enseñado a las máquinas a procesar y generar texto, el siguiente paso en nuestro viaje por la IA nos lleva a otro dominio fundamental de la percepción humana: **la visión**.

En la próxima sección, exploraremos cómo la **Visión por Computadora** está permitiendo a las máquinas *ver* y *comprender* el mundo visual, un campo que, al igual que el NLP, viene

experimentando avances revolucionarios gracias al *deep learning* y está transformando industrias enteras.

08. 👁️ Visión por Computadora (Computer Vision)

Visión por Computadora: Interpretando el Mundo Visual

La visión por computadora analiza imágenes y videos, replicando y superando la capacidad del ojo humano. Es crucial para aplicaciones como conducción autónoma, vigilancia y realidad aumentada.



Uno de los primeros temas que capturó mi atención cuando comencé a especializarme en Inteligencia Artificial fue la capacidad de entrenar modelos que puedan detectar objetos en imágenes o clasificar diferentes imágenes. Naturalmente tuve que hacer algunas pruebas de este estilo 😛[64]:



Who's that Pokémon? The EASIEST Image Classification project using Convolutional Neural Networks with Keras



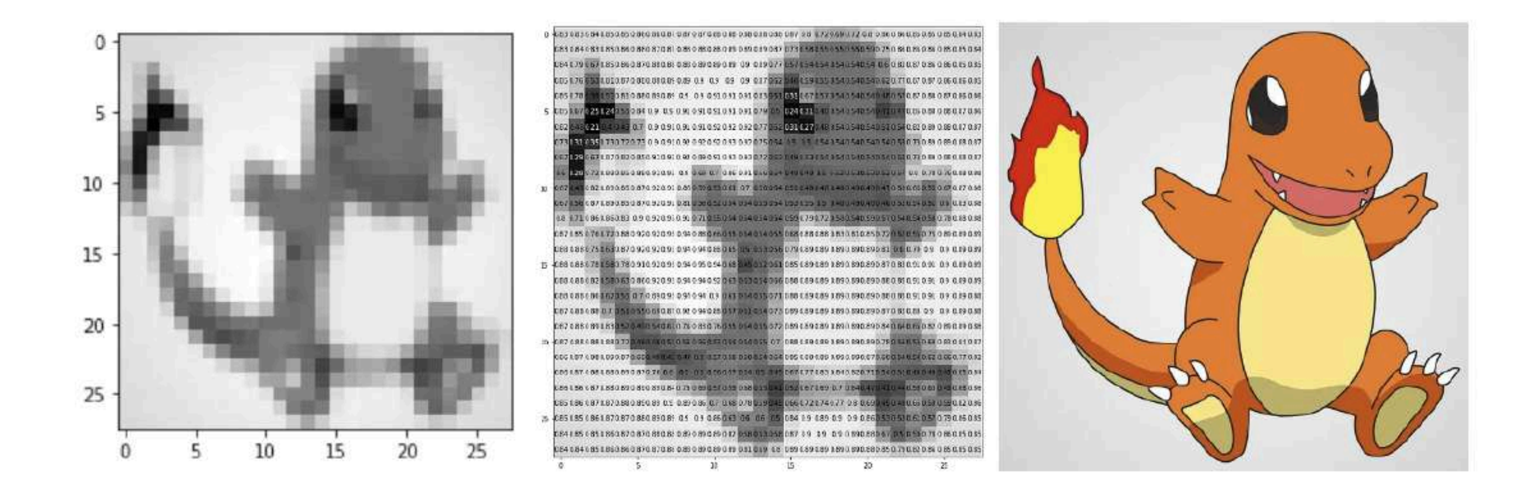
[64] https://medium.com/@gabrielpierobon/whos-that-pok%C3%A9mon-cd02090ab81c

Lo más importante al principio fue entender que una imágen no es más que una grilla de pixeles, cada uno con un valor numérico. En el caso de una imágen de 28 x 28 píxeles como la de abajo, serán 784 valores en total (considerando una imágen en escala de grises) y un modelo puede aprender los patrones numéricos de esa secuencia bastante bien.

Esto es fundamental porque nos ayuda a entender cómo "ve" una computadora: no ve formas ni objetos como nosotros, sino una matriz de números. Por ejemplo:

- Un píxel negro se representa como 0
- Un píxel blanco se representa como 1
- Los valores entre 0.0 y 1.0 representan diferentes tonos de gris

Entonces, una imagen simple se convierte en algo así:



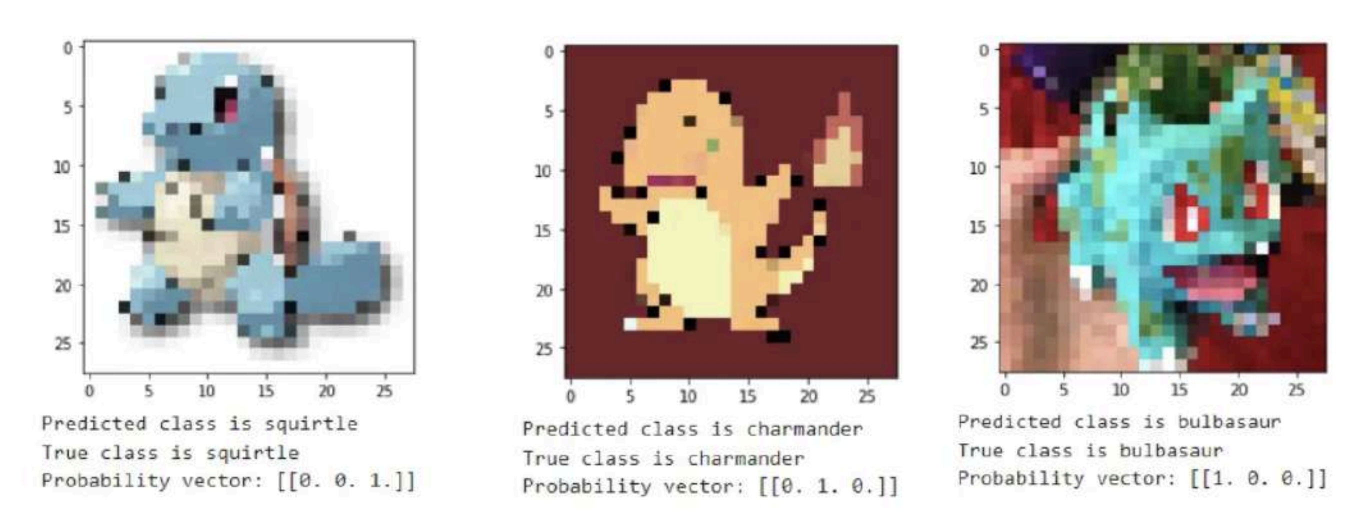
Esta representación numérica es lo que hace posible que los algoritmos de machine learning puedan:

- Detectar bordes y formas
- Reconocer patrones
- Clasificar imágenes
- Identificar objetos

Es similar a cómo procesamos texto: así como convertimos palabras en números para que las máquinas puedan procesarlas, las imágenes ya vienen naturalmente en formato numérico, solo que organizadas en una grilla bidimensional.

Fijate como los píxeles más claros tienen a acercarse a 1.00 y los más oscuros a 0.00.

El modelo que entrené entonces podía predecir qué Pokémon era el de una nueva imágen:

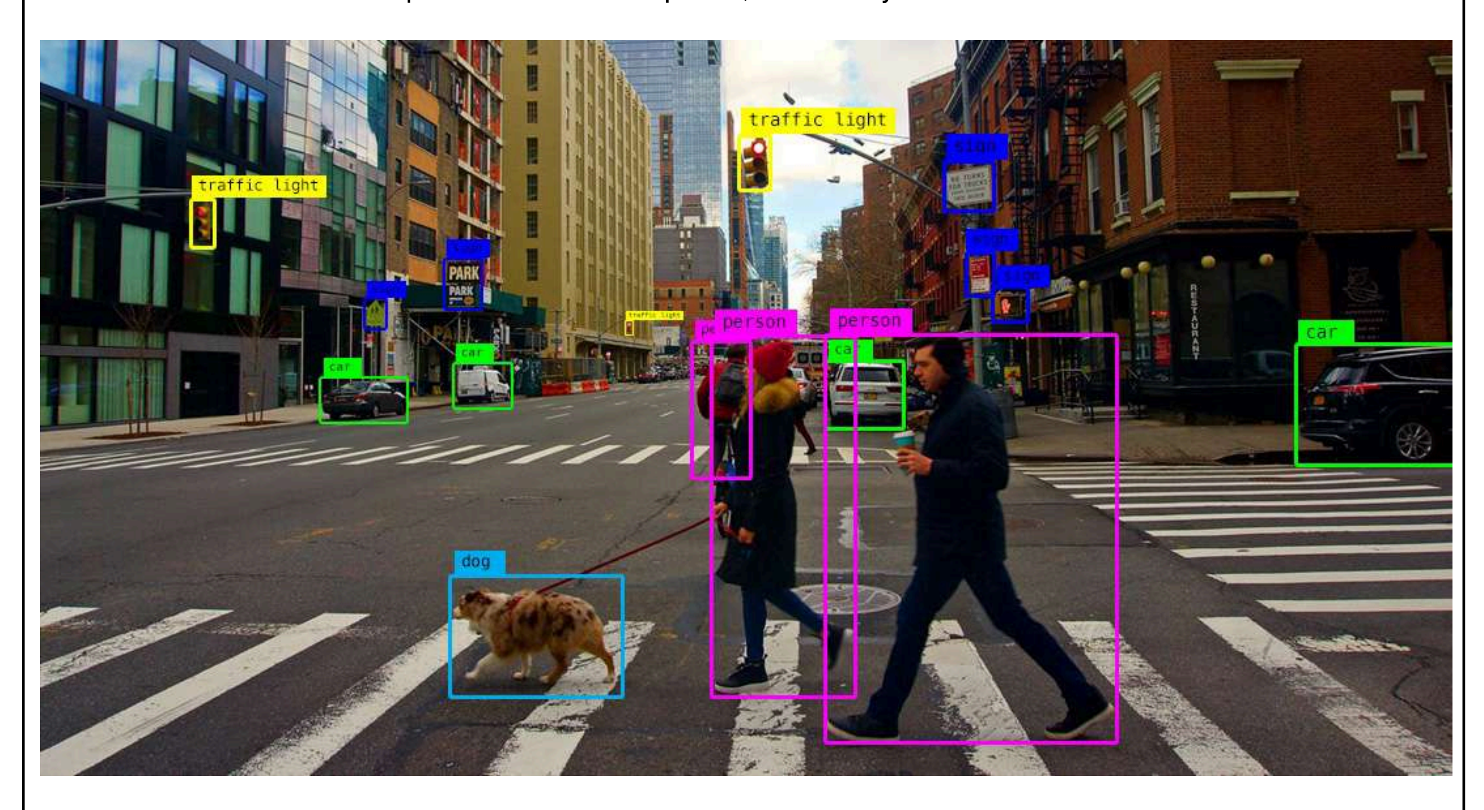


¿Cómo logra esto y mucho más la inteligencia artificial? Veámoslo…

En *Computer Vision* se utilizan técnicas avanzadas para procesar y analizar imágenes y videos, replicando y superando, en algunos casos, la capacidad del ojo humano para reconocer y comprender el entorno visual. La visión por computadora es crucial para una amplia gama de aplicaciones, desde la conducción autónoma hasta la seguridad y la salud.

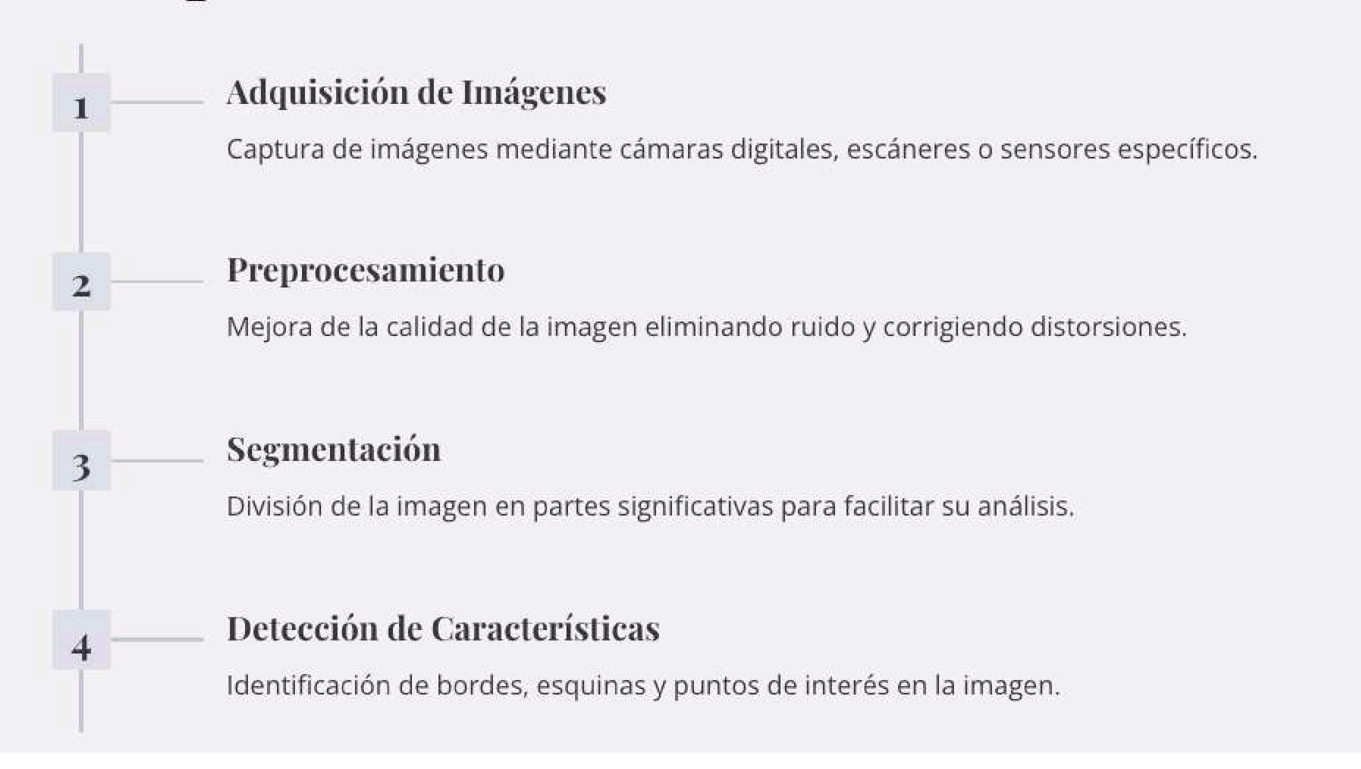
Esta tecnología funciona en varios niveles de complejidad:

- **Nivel básico**: Detectar bordes, colores y formas simples
- **Nivel intermedio**: Reconocer objetos, personas y texto
- **Nivel avanzado**: Interpretar escenas completas, acciones y contextos



La visión por computadora implica la adquisición, procesamiento y análisis de datos visuales (imágenes/videos) para extraer información significativa. Los sistemas de visión por computadora utilizan cámaras, sensores y algoritmos de procesamiento de imágenes para capturar y analizar datos visuales. Este proceso se puede dividir en cuatro etapas fundamentales:

1. **Adquisición de datos**:
2. **Preprocesamiento de imágenes**:
3. **Segmentación**
4. **Reconocimiento (Detección)**

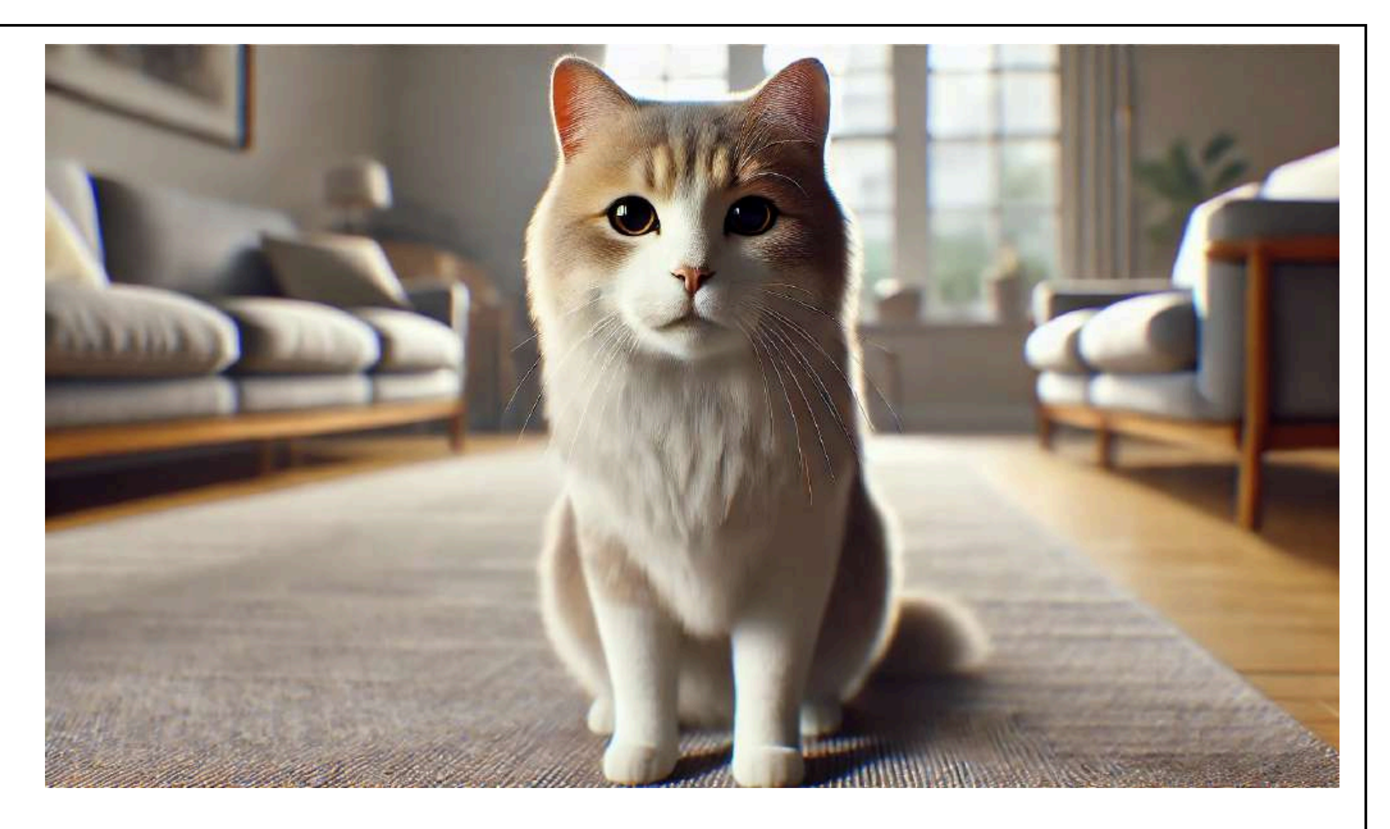


Adquisición de Imágenes: El primer paso en cualquier sistema de visión por computadora es la adquisición de imágenes. Esto se realiza mediante cámaras digitales, escáneres o sensores específicos que capturan imágenes estáticas o videos. Para entrenar modelos y hacer pruebas, podemos encontrar muchísimos datasets online o directamente descargar imágenes públicas.

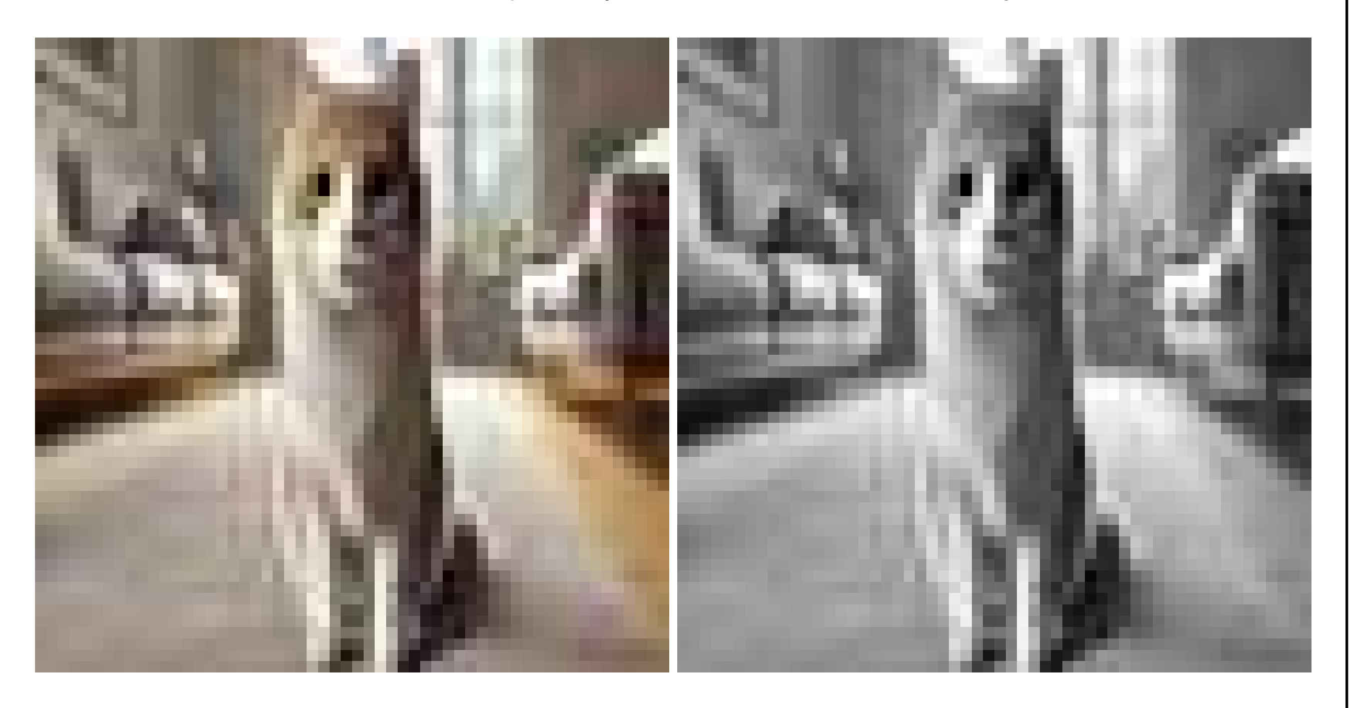
Preprocesamiento de Imágenes: Antes de que una imagen pueda ser analizada, a menudo necesita ser preprocesada. Esto incluye tareas como el redimensionamiento, la eliminación de ruido, la corrección de distorsiones, la detección de contornos y el ajuste de contraste para mejorar la calidad de la imagen.

- El **redimensionamiento** (escalado) de imágenes es el proceso de cambiar el tamaño de una imagen, ya sea aumentándolo o reduciéndolo. Este paso es crucial para estandarizar el tamaño de las imágenes antes de procesarlas, ya que muchos algoritmos y redes neuronales requieren que todas las imágenes de entrada tengan las mismas dimensiones. Por ejemplo, si tenemos fotos de diferentes tamaños (800x600, 1024x768, etc.) y nuestro modelo espera imágenes de 56x56 píxeles, necesitamos escalar todas las imágenes a este tamaño uniforme. Sin embargo, el escalado puede afectar la calidad de la imagen: al reducir el tamaño podemos perder detalles, y al aumentarlo podemos crear pixelación o borrosidad, por lo que se utilizan diferentes técnicas de interpolación para mantener la mejor calidad posible.

  A continuación veamos un ejemplo:



Redimensionamiento a 56 x 56 a la izquierda y adicionalmente escalado de grises, a la derecha.



La conversión de una imagen a escala de grises reduce su complejidad al pasar de tres canales de color (RGB) a un único canal de intensidad. Esto no solo disminuye el costo computacional del procesamiento, sino que también simplifica muchos algoritmos de análisis que no requieren información de color. Es como convertir una fotografía a color en una foto en blanco y negro, donde cada píxel representa únicamente qué tan brillante u oscuro es ese punto.

- **Ajuste de brillo y contraste**: Estos ajustes mejoran la visibilidad de los detalles en la imagen. El ajuste de brillo modifica la intensidad general de la imagen, haciendo que sea más clara u oscura, mientras que el contraste aumenta la diferencia entre las áreas claras y oscuras. Es similar a cuando

ajustamos la pantalla de nuestro teléfono para ver mejor bajo la luz solar: estamos modificando estos parámetros para que la información sea más visible.

- **Detección de contornos**: Los contornos o bordes son cambios abruptos en la intensidad de la imagen que generalmente corresponden a los límites de los objetos. La detección de bordes identifica estos cambios, creando un "esqueleto" de la imagen que resalta las formas y estructuras principales. Es similar a hacer un dibujo a línea de una fotografía, donde sólo mantenemos los contornos principales.

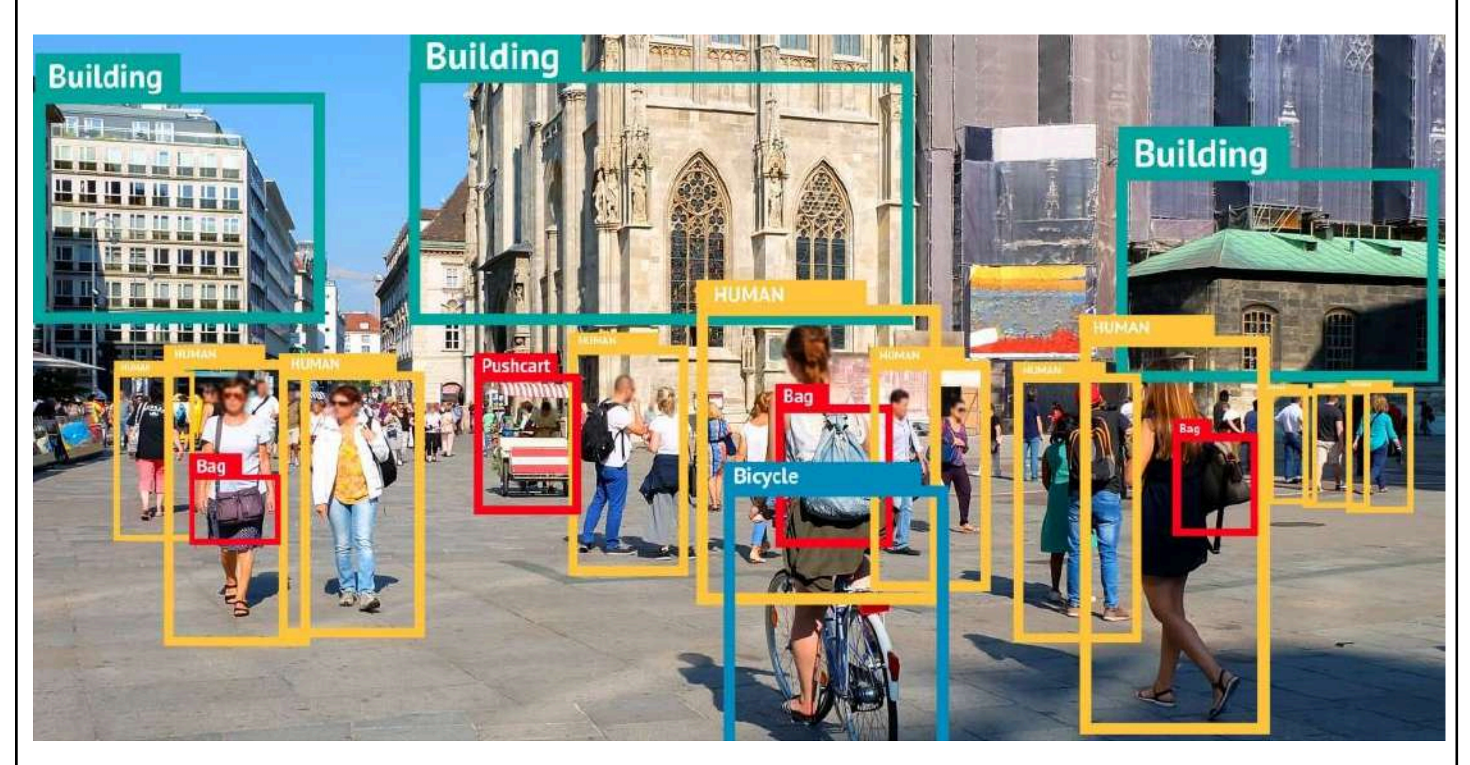


Estas técnicas de preprocesamiento tienen como objetivo ayudar a nuestro modelo a tener una tarea más sencilla y enfocarse en lo que realmente es importante. En el caso de la detección de imágenes de gatos, es fácil de entender que transformar las imágenes a una detección de contornos puede ser de extrema utilidad para que el modelo se enfoque en la detección de lo que realmente es un gato y librarse de todo el resto de información de poco valor de la imágen.

Segmentación de imágenes: La segmentación divide una imagen en partes o regiones significativas para facilitar su análisis. Es como cuando miramos una fotografía grupal y nuestro cerebro automáticamente "recorta" la imagen separando cada persona, el fondo, y los objetos importantes. Por ejemplo, en una imagen de una calle, la segmentación puede separar los coches, los peatones y los edificios, tal como nosotros naturalmente distinguimos estos elementos cuando miramos la escena. Este proceso es fundamental porque permite que el sistema "entienda" dónde termina un objeto y dónde empieza otro, similar a cuando un nene aprende a distinguir que su juguete es un objeto separado de la mesa sobre la que está apoyado. En aplicaciones prácticas, esto permite que un coche autónomo sepa exactamente dónde están los límites de la carretera, dónde están los otros vehículos y dónde están los peatones, tratando cada elemento como una entidad separada que requiere un análisis específico.

Reconocimiento (Detección): A diferencia de la segmentación que nos dice "dónde" está cada elemento en una imagen, el reconocimiento se encarga de identificar y etiquetar "qué" es cada objeto. Es similar a cómo

nosotros, al entrar a una habitación, automáticamente reconocemos "acá hay una silla", "allá una mesa", "esa es una lámpara", sin necesidad de pensarlo conscientemente. El sistema aprende a reconocer patrones y características distintivas de cada objeto, tal como nosotros aprendemos desde chiquitos que una silla típicamente tiene cuatro patas y un respaldo. Por ejemplo, cuando el sistema analiza una foto de una cocina, puede identificar "esto es un horno" porque aprendió que los hornos suelen ser rectangulares, metálicos y estar ubicados contra la pared, "esto es una taza" porque reconoce su forma característica con un asa, y "esto es una fruta" por su color y forma específica. Esta capacidad es fundamental para numerosas aplicaciones, desde sistemas de seguridad que detectan objetos sospechosos hasta aplicaciones de compras que pueden identificar productos en fotos.



Las técnicas que acabamos de ver (preprocesamiento, segmentación y reconocimiento de objetos) son fundamentales para entender cómo se entrena un modelo de clasificación de imágenes. Así como nosotros necesitamos que alguien nos muestre muchos ejemplos de "gatos" y "perros" para aprender a diferenciarlos cuando somos niños, los modelos necesitan "ver" miles de imágenes etiquetadas para aprender a reconocer patrones. El preprocesamiento asegura que todas estas imágenes de entrenamiento estén en un formato consistente y óptimo para el aprendizaje, similar a cómo un profesor aseguraría que todos los materiales educativos sean claros y legibles para sus estudiantes. La segmentación y el reconocimiento nos ayudan a entender qué características son importantes para la clasificación, como la forma de las orejas en un gato o el hocico en un perro. Veamos ahora cómo se combinan todos estos elementos en el proceso de entrenamiento de un modelo de clasificación.

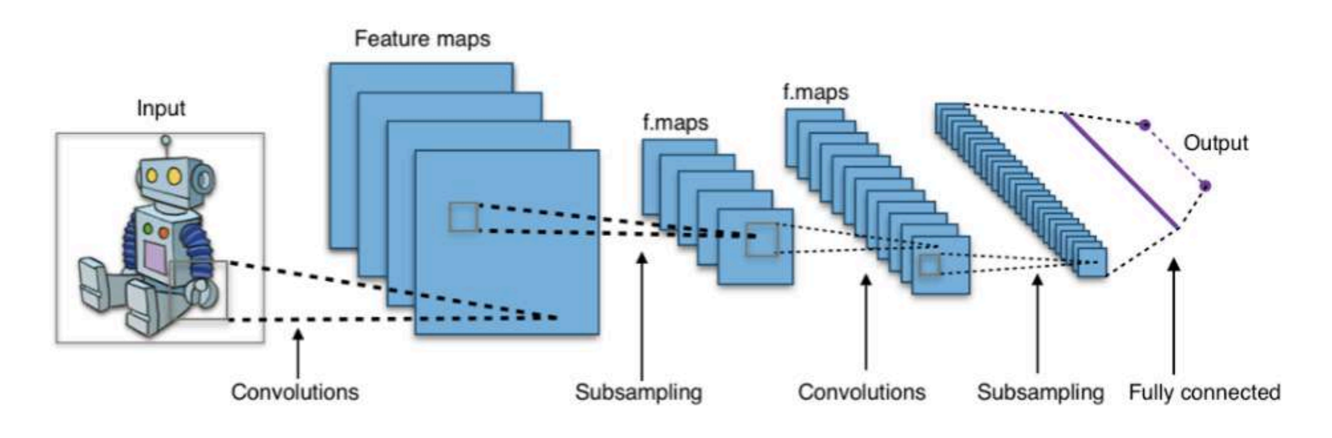
¿Cómo se entrena un modelo para clasificación de imágenes?

Supongamos que querés desarrollar un sistema de inteligencia artificial capaz de clasificar imágenes en distintas categorías. Este proceso, conocido como *clasificación de imágenes*, es fundamental en muchas aplicaciones de visión por computadora, desde el reconocimiento de rostros hasta el diagnóstico médico automatizado.

El primer paso como siempre se trata de la **recopilación y preparación de los datos**. Vas a necesitar un conjunto de datos grande, que pueda tener hasta miles de imágenes, cada una etiquetada con su categoría correspondiente. Estas imágenes, como ya vimos en el inicio de esta sección, se van a representar como

matrices tridimensionales[65] de números, donde cada número representa la intensidad del color en un punto específico (píxel) de la imagen. Por ejemplo, una imagen a color de 224x224 píxeles se transforma en una matriz de 224x224x3, donde el 3 representa los canales de color rojo, verde y azul (RGB).

Como vimos en nuestra sección sobre *redes neuronales*, el corazón del sistema de clasificación de imágenes es una **red neuronal *convolucional* (CNN)[66]**. Esta arquitectura está inspirada en el córtex visual del cerebro humano[67] y es especialmente eficaz para procesar datos con estructura de *cuadrícula*, como las imágenes. Las CNN constan de varias capas:



[68]

La imagen de arriba representa el funcionamiento de una red neuronal convolucional (CNN) con los componentes que hemos descrito. El proceso comienza con una imagen de entrada, en este caso un dibujo de un robot. Esta imagen pasa por varias capas de la red. Primero, las capas convolucionales extraen características de la imagen, desde simples (bordes, esquinas) hasta complejas (ojos, pies, manos). Luego, las capas de subsampling o pooling reducen el tamaño de los datos manteniendo la información esencial. Este proceso de convolución y submuestreo se repite varias veces, creando mapas de características cada vez más abstractos. Finalmente, los datos pasan por una o más capas completamente conectadas que integran todas las características para producir una salida, que podría ser una clasificación o predicción sobre la imagen original. Este diseño permite a la red aprender automáticamente a reconocer patrones relevantes en las imágenes, mejorando su capacidad para tareas como la clasificación de objetos o el reconocimiento facial.

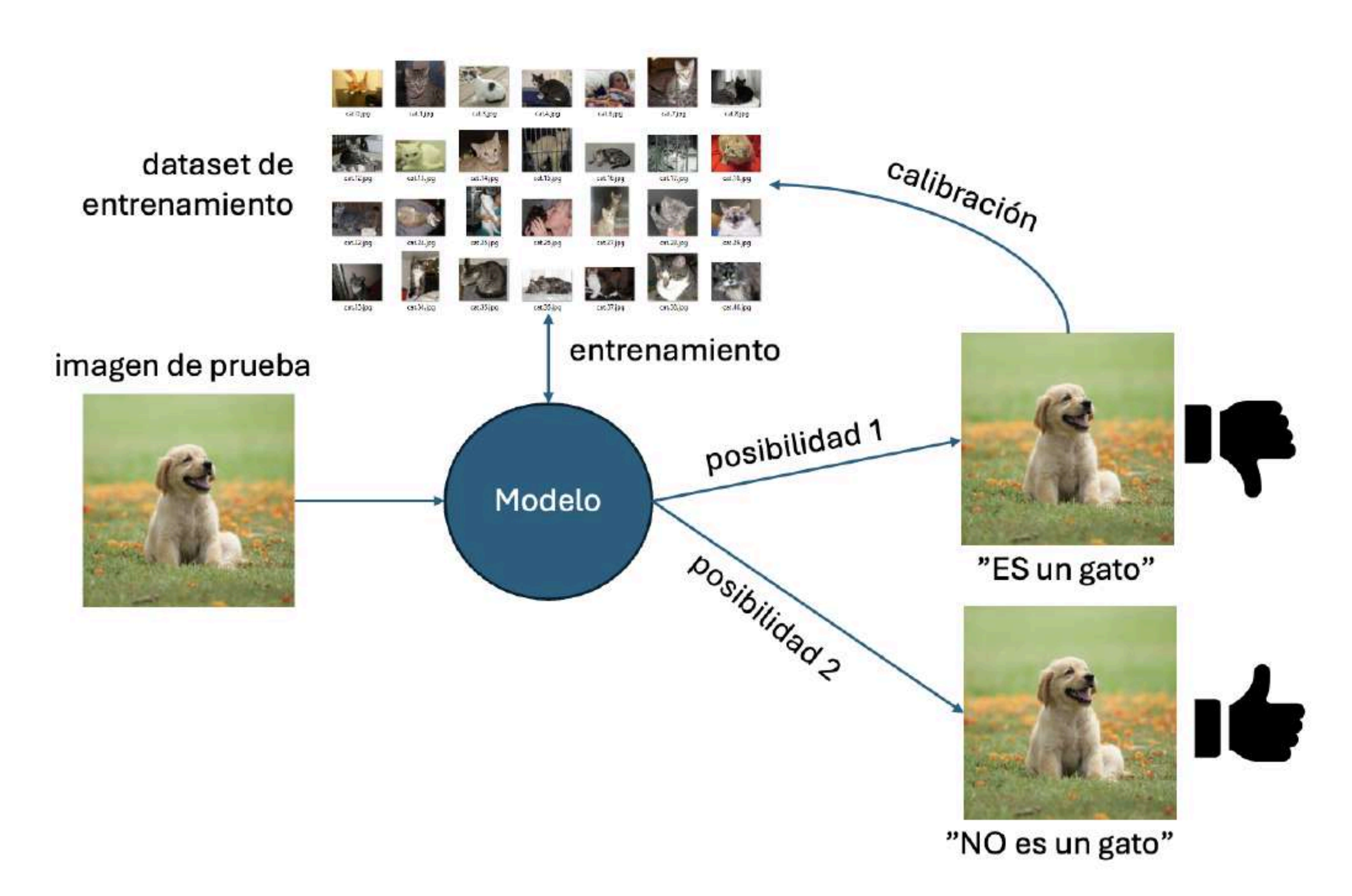
- **Capas convolucionales (Convolutions)**: Actúan como *detectores de características*, aplicando filtros que aprenden a reconocer patrones específicos en la imagen (bordes, texturas, formas). Estos filtros se deslizan sobre la imagen, realizando operaciones matemáticas que permiten detectar la presencia de características particulares en diferentes ubicaciones. Por ejemplo, un filtro podría especializarse en detectar bordes verticales, mientras que otro podría identificar texturas rugosas. Es similar a cómo nuestros ojos escanean una imagen enfocándose en diferentes detalles cada vez - primero podríamos notar los bordes de un objeto, luego su textura, después su forma general.
- **Capas de agrupación (pooling)**: Reducen la dimensionalidad de los datos, conservando las características más importantes. Típicamente utilizando operaciones como "max pooling" o "average pooling", estas capas dividen la imagen en regiones y seleccionan o promedian los valores más

[65] En nuestro ejemplo inicial, la matriz era bidimensional por que al transformar la imágen a escala de grises nos quitamos la dimensión del color.
[66] Si bien existen otros algoritmos capaces de clasificar imágenes como Support Vector Machines (SVM), Random Forests o incluso redes neuronales tradicionales (fully connected), las CNN se han convertido en el *gold standard* para el procesamiento de imágenes debido a su capacidad de aprender automáticamente características relevantes y su eficiencia computacional al procesar datos visuales.
[67] El córtex visual es la parte del cerebro responsable de procesar la información visual. Está organizado en capas jerárquicas donde cada capa detecta características cada vez más complejas: primero detectamos bordes simples, luego formas básicas, y finalmente objetos completos. Las CNN imitan esta organización jerárquica, lo que las hace tan efectivas para procesar imágenes.
[68] https://milliams.com/courses/neural_networks/Convolutional%20neural%20networks.html de acá saqué la imágen

significativos de cada región. Esto no solo reduce el tamaño de los datos a procesar, sino que también hace al modelo más robusto ante pequeñas variaciones en la posición de los objetos. Es como cuando miramos una foto desde lejos y seguimos reconociendo los objetos principales aunque no veamos todos los detalles.

- **Capas de submuestreo (Subsampling)**: Estas capas reducen progresivamente la resolución espacial de las características extraídas, permitiendo que la red capture información a diferentes escalas. El submuestreo ayuda a reducir la cantidad de parámetros y cálculos necesarios, mejorando la eficiencia computacional mientras mantiene la capacidad de la red para reconocer patrones importantes. A diferencia del pooling, el submuestreo puede realizarse mediante convoluciones con "stride" mayor a 1, lo que permite que la red aprenda la mejor manera de reducir la dimensionalidad. Es similar a cómo nuestro cerebro puede reconocer un objeto tanto en una foto pequeña como en una grande, adaptándose a diferentes escalas.
- **Capas completamente conectadas (Fully connected)**: Toman las características extraídas y las utilizan para la clasificación final. Estas capas funcionan como un clasificador tradicional, donde cada neurona está conectada con todas las neuronas de la capa anterior. Acá es donde la red combina todas las características detectadas para tomar una decisión sobre la clase a la que pertenece la imagen, asignando probabilidades a cada posible categoría. Es como cuando nuestro cerebro junta todas las pistas visuales (forma, color, textura) para decidir "esto definitivamente es un gato".

¿Cómo se *entrena* el modelo? Ya lo vimos en nuestra sección de *redes neuronales* pero volvemos a repasarlo.



El **proceso de entrenamiento** implica alimentar estas imágenes a través de la red, inicialmente configurada con pesos aleatorios (no nos importa la configuración inicial por que sabemos que con la optimización estos pesos se irán ajustando). La red hace una predicción, que se compara con la *etiqueta* real de la

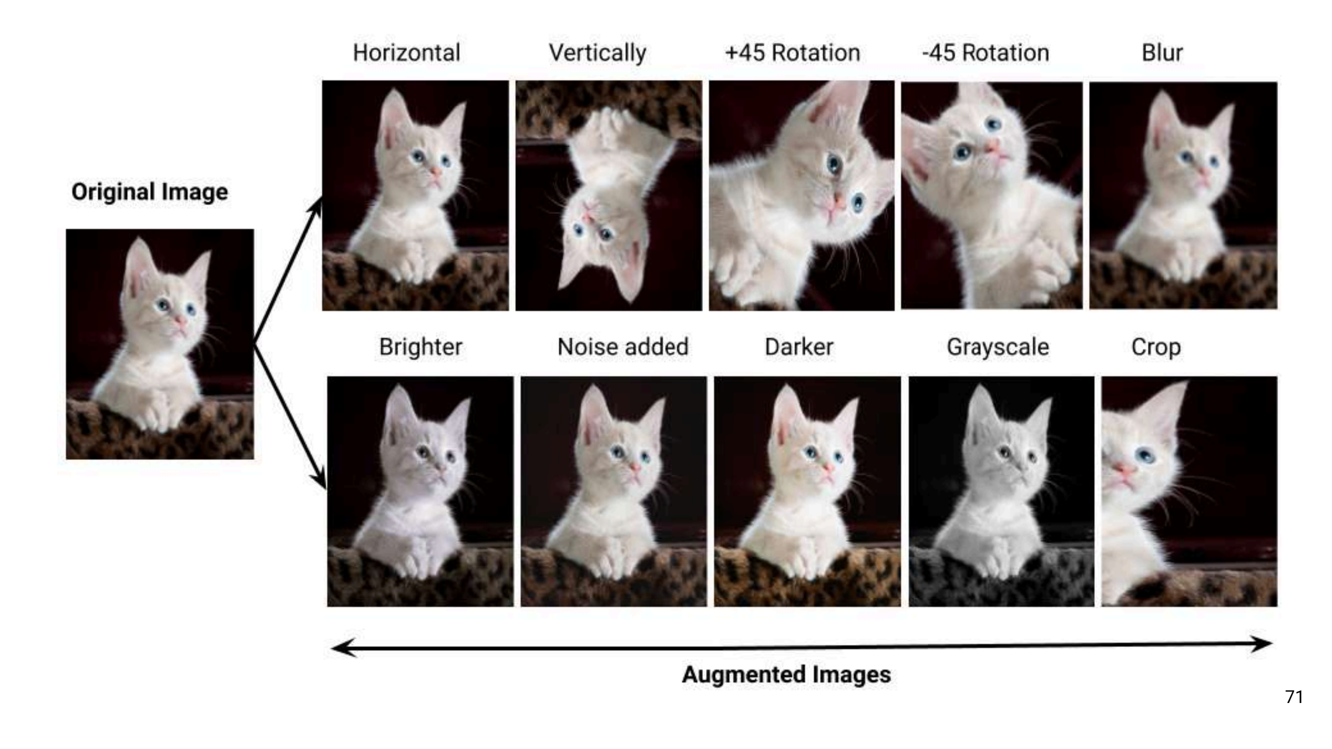
imagen. La diferencia entre la predicción y la realidad (el error) se utiliza para ajustar los pesos de los parámetros de la red mediante el proceso de *backpropagation* que ya conocés. Este ajuste se realiza utilizando **algoritmos de optimización** como el *descenso de gradiente estocástico (SGD)*.

El entrenamiento en realidad no se hace imágen por imágen, sino que se realiza en *lotes* (*batches*) de imágenes, y cada *pasada completa* a través del conjunto de datos se llama *época* (lo vimos en la sección de *deep learning* con el ejemplo de Google). A medida que avanzan las *épocas*, la red aprende a extraer características cada vez más complejas y relevantes. Las primeras capas de la red típicamente podrían aprender a detectar bordes y formas simples, mientras que las capas más profundas podrían reconocer características más abstractas y específicas de cada clase.

Escribí un artículo completo sobre este proceso hace algunos años[69].

Para evitar el *sobreajuste* (*overfitting*), donde el modelo *memoriza* el conjunto de datos de entrenamiento en lugar de aprender a *generalizar*, se emplean técnicas de *regularización*[70]. Estas incluyen la *normalización por lotes* (*batch normalization*), el *dropout* (desactivación aleatoria de neuronas durante el entrenamiento), y la *aumentación de datos* (*data augmentation*), donde se crean variaciones artificiales de las imágenes de entrenamiento mediante rotaciones, recortes o cambios de iluminación.

En la siguiente imágen podemos ver cómo a partir de una única imágen podemos crear una variedad de la misma aplicando diferentes técnicas de *data augmentation*. Esto para nosotros debería resultar redundante y hasta innecesario, pero para la *red neuronal CNN* puede permitirle captar patrones que con la imágen original podría omitir.



[71]

[69] https://medium.com/towards-data-science/visualizing-intermediate-activation-in-convolutional-neural-networks-with-keras-260b36d60d0

[70] Recordá que en IA y ciencia de datos no nos interesa un modelo que es muy bueno con los datos de entrenamiento pero malo con nuevos datos similares. De la misma forma que en educación no es recomendado memorizar respuestas a preguntas de un cuestionario sino aprender el tema para poder responder otras preguntas parecidas pero no iguales.

[71] https://ubiai.tools/what-are-the-advantages-anddisadvantages-of-data-augmentation-2023-update/

El *rendimiento* del modelo (qué tan útil es) se evalúa con un conjunto de *datos de validación* separado[72], que el modelo nunca vió durante el entrenamiento. Métricas como la *precisión*, el *recall* y el *F1-score* se utilizan para medir qué tan bien el modelo generaliza a nuevos datos. Si el rendimiento en el conjunto de validación es satisfactorio, se realiza una evaluación final en un conjunto de prueba completamente independiente para confirmar la capacidad de generalización del modelo.

Finalmente, el modelo entrenado puede desplegarse en aplicaciones del mundo real, donde puede clasificar nuevas imágenes con alta precisión. Sin embargo, el proceso no termina ahí: el modelo debe ser monitoreado y reentrenado periódicamente con nuevos datos para mantener su rendimiento a lo largo del tiempo, especialmente en entornos dinámicos donde la distribución de los datos puede cambiar.

Aprovechamos que hemos introducido estos tópicos importantes del mundo de *clasificación* que no se han incluído en la sección de *machine learning supervisado* y los vemos un poco más en detalle:

Precisión, recall y F1-Score en clasificación (de imágenes)

Imaginá que tenés un modelo de IA que clasifica fotos de animales. Para saber qué tan bien funciona, usamos tres métricas principales: *precisión*, *recall* y *F1-score*. Estas métricas nos ayudan a entender diferentes aspectos del rendimiento del modelo.

Precisión

La precisión responde a la pregunta: *"De todas las veces que el modelo dijo que una imagen era de un perro, ¿cuántas veces acertó realmente?"*

Fórmula: **Precisión = Verdaderos Positivos / (Verdaderos Positivos + Falsos Positivos)**

Ejemplo: Si el modelo clasificó 100 imágenes como perros, pero solo 90 eran realmente perros, la precisión sería 90/100 = 0.90 o 90%.

Una alta precisión significa que cuando el modelo identifica algo, podés confiar bastante en que está en lo correcto.

Recall (Exhaustividad)

El recall responde a la pregunta: *"De todos los perros que había en las todas imágenes, ¿cuántos logró identificar correctamente el modelo?"*

Fórmula: **Recall = Verdaderos Positivos / (Verdaderos Positivos + Falsos Negativos)**

Ejemplo: Si había 150 imágenes de perros en total, pero el modelo solo identificó correctamente 120, el recall sería 120/150 = 0.80 o 80%.

Un alto recall significa que el modelo es bueno para encontrar todos los ejemplos de una clase, aunque a

[72] Es decir que al comenzar un proyecto como este, creamos tres tipos de conjuntos de imágenes. El conjunto de entrenamiento (tipicamente el 70% del total) con el que se entrena el modelo. El conjunto de test/prueba (por ejemplo el 20% del total) con el cual el modelo evalúa el rendimiento durante el entrenamiento y determina hasta cuándo debe continuar el proceso de optimización. Finalmente debemos dejar un conjunto de validación para el final del proceso (un 10% de las imágenes totales) para medir y documentar el rendimiento final del modelo. Ya que tanto el conjunto de entrenamiento como el de prueba, técnicamente participan del proceso de entrenamiento, es el conjunto de validación el único que replica las condiciones reales del modelo enfrentándose a nuevos datos nunca antes vistos ni considerados. Es posible que en muchos proyectos de ciencia de datos veas que solo se trabaja con conjuntos de entrenamiento y de test, y puede que bajo determinadas circunstancias eso esté bien, pero el proceso comúnmente aceptado es el descrito en este apartado.

veces pueda equivocarse.

F1-Score

El F1-score es como un promedio entre la precisión y el recall. Es útil cuando querés un solo número que resuma qué tan bien funciona tu modelo, especialmente cuando tenés un desequilibrio en tus clases (por ejemplo, muchas más fotos de perros que de gatos).

Fórmula: **F1 = 2 * (Precisión * Recall) / (Precisión + Recall)**

El F1-score será alto solo si tanto la precisión como el recall son altos.

Cómo se usan en la práctica

- **Evaluación** del modelo: Calculás estas métricas en tu conjunto de prueba para ver qué tan bien funciona tu modelo en general.
- **Comparación** de modelos: Podés usar estas métricas para comparar diferentes versiones de tu modelo o diferentes modelos entre sí.
- **Ajuste** del modelo: Dependiendo de tu aplicación, podés decidir si es más importante tener una alta precisión o un alto recall, y ajustar tu modelo en consecuencia.
- **Detección de problemas**: Si ves que tu precisión es alta pero tu recall es bajo (o viceversa), te da pistas sobre qué tipo de errores está cometiendo tu modelo.
- **Monitoreo** en producción: Una vez que tu modelo está en uso, podés seguir calculando estas métricas para asegurarte de que sigue funcionando bien con nuevos datos.

Equilibrio entre precisión y recall: ¿Qué priorizar?

La elección entre priorizar precisión o recall depende fundamentalmente del **costo de los errores** en cada contexto específico. Consideremos algunos ejemplos:

Casos donde priorizar alta *precisión*:

- **Sistemas de recomendación de productos**: Es preferible mostrar menos recomendaciones pero más acertadas, que inundar al usuario con sugerencias irrelevantes. Un falso positivo (recomendar un producto inadecuado) puede reducir la confianza del usuario en el sistema.
- **Filtros de spam**: Es mejor dejar pasar algunos correos spam al inbox que bloquear correos legítimos importantes. Un falso positivo (bloquear un email válido) podría tener consecuencias significativas.

Casos donde priorizar alto *recall*:

- **Detección de enfermedades**: Es preferible tener algunos falsos positivos que requieran pruebas adicionales, que perder la detección de una enfermedad grave. El costo de un falso negativo (no detectar una enfermedad presente) puede ser crítico.
- **Sistemas de seguridad**: En la detección de actividades fraudulentas o amenazas, es mejor generar algunas alertas falsas que perder la detección de una amenaza real. Un falso negativo podría resultar en una brecha de seguridad.

El *F1-score* se vuelve especialmente relevante cuando necesitamos un balance entre ambas métricas, como en sistemas de clasificación de imágenes para moderación de contenido, donde tanto los falsos positivos como los falsos negativos tienen costos significativos.

En el contexto específico de la visión por computadora, la elección entre precisión y recall a menudo depende del tipo de aplicación que estemos desarrollando. Por ejemplo, en sistemas de conducción

autónoma, necesitamos un alto recall en la detección de peatones y obstáculos - es preferible que el coche se detenga innecesariamente algunas veces (falsos positivos) a que no detecte a un peatón (falso negativo). En cambio, en aplicaciones de reconocimiento facial para desbloqueo de dispositivos, priorizamos la precisión - es mejor que el usuario tenga que intentar desbloquear su teléfono más de una vez (falso negativo) a que cualquier rostro similar pueda desbloquearlo (falso positivo). En sistemas de control de calidad industrial que inspeccionan productos mediante cámaras, el balance entre precisión y recall debe ajustarse según los costos: ¿es más costoso desechar un producto bueno o dejar pasar uno defectuoso? Estas consideraciones específicas del dominio visual deben guiar nuestras decisiones al desarrollar y ajustar modelos de visión por computadora.

Aplicaciones del *Computer Vision*

🚗 **Conducción autónoma**: Los vehículos autónomos utilizan visión por computadora para interpretar su entorno, identificar objetos como otros vehículos, peatones y señales de tráfico, y tomar decisiones de conducción en tiempo real.



[73]

🔒 **Seguridad y vigilancia**: Las cámaras de seguridad equipadas con sistemas de visión por computadora pueden detectar actividades sospechosas, reconocer rostros y monitorizar áreas en tiempo real, mejorando la seguridad en diversas instalaciones.

[73] https://bernardmarr.com/how-tesla-is-using-artificial-intelligence-to-create-the-autonomous-cars-of-the-future/



[74]

🏥 **Salud y medicina**: En el campo de la salud, la visión por computadora se utiliza para analizar imágenes médicas como radiografías, resonancias magnéticas y tomografías, ayudando en el diagnóstico de enfermedades y el seguimiento del progreso de tratamientos.

[75]

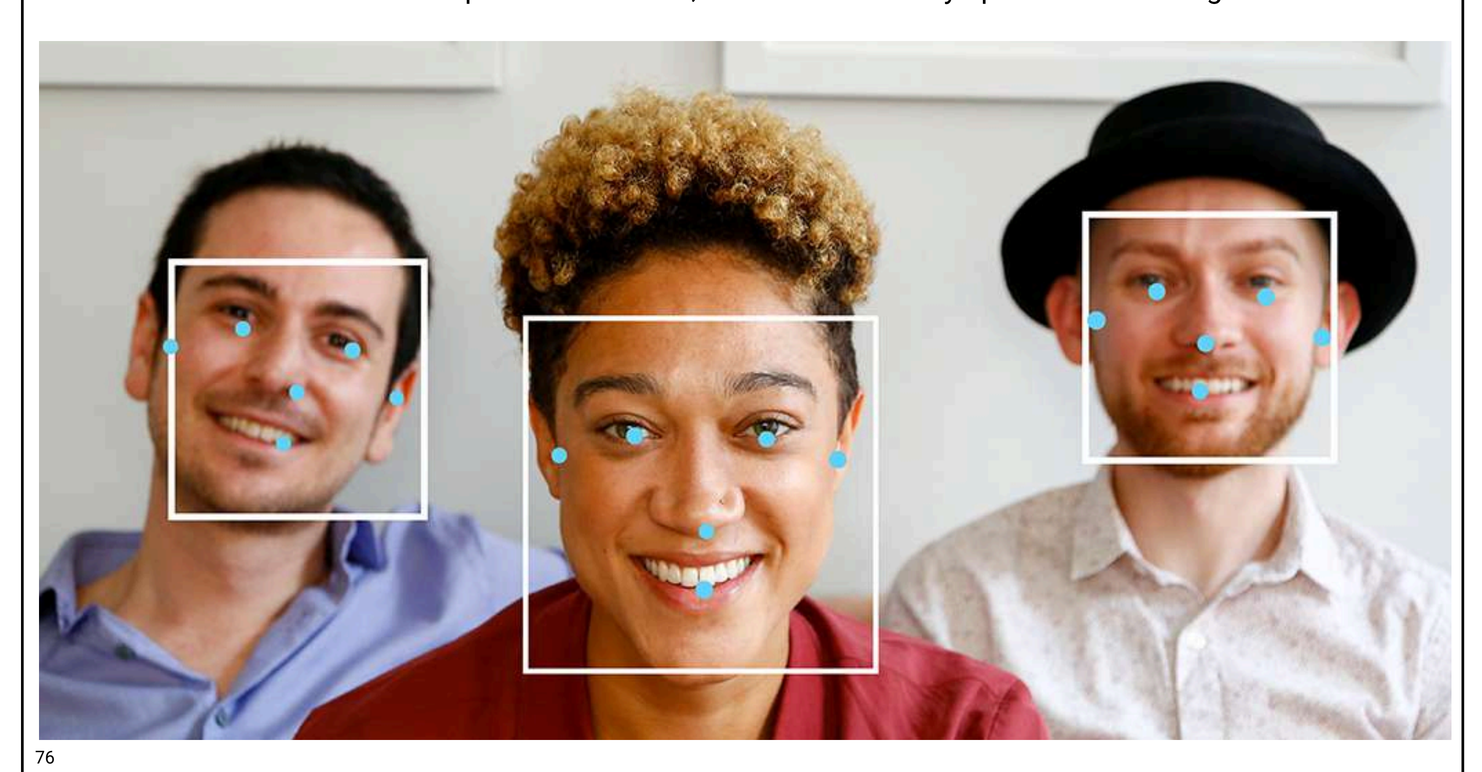
👤 **Reconocimiento facial**: Los sistemas de reconocimiento facial son ampliamente utilizados para la

---

[74] https://www.cctvcamerapros.com/AI-security-cameras-s/1512.htm

[75] https://www.itnonline.com/content/ai-fight-against-covid-19-automatic-detection-chest-x-ray-images-possible-reports-incheon

autenticación de identidad en dispositivos móviles, control de acceso y aplicaciones de seguridad.



76

🕶️ **Realidad aumentada (AR) y realidad virtual (VR)**: La visión por computadora permite la creación de experiencias inmersivas en AR y VR, integrando objetos virtuales en el mundo real y permitiendo interacciones más naturales.



77

🌾 **Agricultura de precisión**: Los drones y sensores equipados con sistemas de visión por computadora se utilizan para monitorear cultivos, detectar enfermedades en plantas y optimizar el uso de recursos en la

[76] https://ai.google.dev/edge/mediapipe/solutions/vision/face_detector
[77] https://www.apple.com/la/newsroom/2023/06/introducing-apple-vision-pro/

agricultura.

78

🛒 **Comercio y retail**: Las tiendas utilizan cámaras y visión por computadora para analizar el comportamiento de los clientes, gestionar el inventario y mejorar la experiencia de compra.



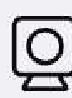
79

[78] https://thedailyguardian.com/others/artificial-intelligence-and-machine-learning-in-agriculture/
[79] https://alwaysai.co/solutions/retail



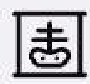
Conducción Autónoma

Interpretación del entorno y toma de decisiones en tiempo real.



Seguridad y Vigilancia

Detección de actividades sospechosas y reconocimiento facial.



Salud y Medicina

Análisis de imágenes médicas para diagnóstico y seguimiento.

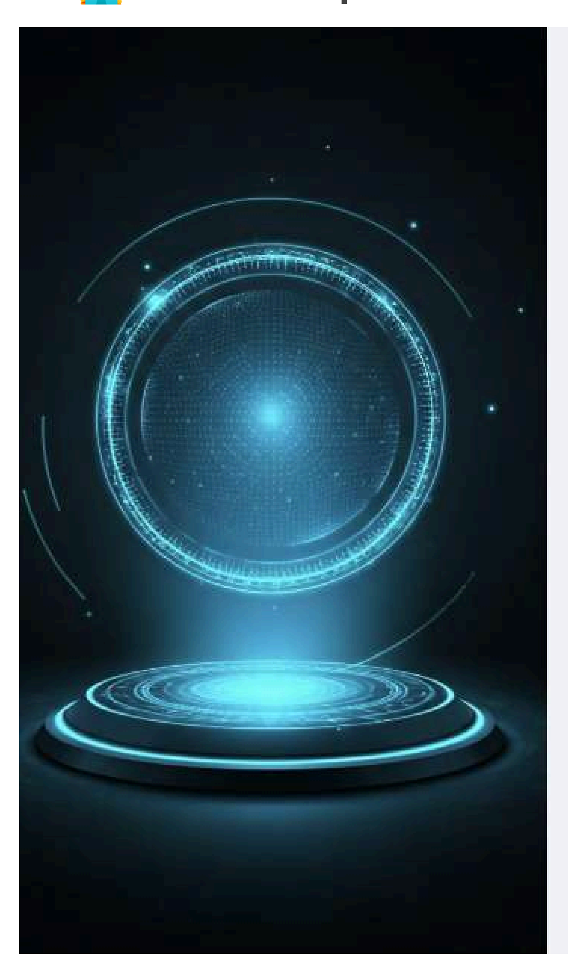


Agricultura de Precisión

Monitoreo de cultivos y optimización de recursos.

Con esto concluímos nuestra sección sobre *visión por computadora*. Hasta ahora, los conceptos básicos de la IA que hemos visto nos han llevado por un recorrido desde los fundamentos lógicos de la IA (con la lógica proposicional y de primer orden), pasando por los diferentes paradigmas del *machine learning* (supervisado, no supervisado y por refuerzo), hasta llegar a las aplicaciones más avanzadas en deep learning, procesamiento de lenguaje natural y visión por computadora. Estos enfoques comparten una característica común: la capacidad de aprender patrones directamente de los datos. Sin embargo, no toda la inteligencia artificial se basa en el *machine learning*. Las siguientes secciones nos introducen en metodologías complementarias que se apoyan en el conocimiento estructurado y el razonamiento probabilístico. Los *sistemas expertos* y las *redes bayesianas* representan un enfoque diferente pero igualmente valioso: la codificación explícita del conocimiento humano y las relaciones de causalidad. Esta transición nos va a permitir apreciar cómo la IA moderna puede beneficiarse tanto del poder del *machine learning* como de la precisión del conocimiento experto codificado, ofreciendo soluciones más robustas y explicables para problemas complejos.

09. 🧑‍💻 Sistemas Expertos



Sistemas Expertos: Emulando la Experiencia Humana

Los sistemas expertos emulan la toma de decisiones de expertos humanos en dominios específicos. Son cruciales para aplicaciones que requieren conocimiento especializado, como diagnóstico médico y asesoría financiera.

Primero que nada te reto a que juegues un juego online para que entiendas a la perfección lo poderoso que puede ser un buen sistema experto.

https://es.akinator.com/

El juego en cuestión es conocido como *Akinator* y es un *sistema experto* en forma de juego web y aplicación móvil que adivina *personajes*. Funciona haciendo preguntas de *sí* o *no* al jugador sobre un personaje que el propio jugador ha pensado. Akinator procesa las respuestas, navega por su extensa base de conocimientos y aplica reglas de inferencia para reducir las posibilidades y adivinar el personaje. Su eficacia se debe a su capacidad para manejar una gran cantidad de información, procesar respuestas ambiguas y aprender de nuevos casos, permitiéndole adivinar incluso personajes poco conocidos con precisión sorprendente.

Si no lo conocías, te va a impresionar! Y si no te impresiona tanto al menos te va a dar una noción sobre los sistemas expertos de los que vamos a empezar a hablar ahora.

Los sistemas expertos utilizan conocimientos codificados y reglas para resolver problemas complejos, proporcionando recomendaciones, diagnósticos o decisiones en áreas como la medicina, la ingeniería, las finanzas y más. Los sistemas expertos fueron una de las primeras aplicaciones prácticas de la inteligencia artificial y siguen siendo una herramienta valiosa para capturar y utilizar el conocimiento humano en diversas industrias[80].

Fundamentos de los Sistemas Expertos

💡 **Base de conocimiento**: El corazón de un sistema experto es su *base de conocimiento*, que contiene hechos y reglas sobre un dominio específico. Por ejemplo, en un sistema experto para recomendar inversiones financieras, la base de conocimiento contendría hechos como "el cliente tiene un perfil conservador" o "la tasa de interés actual es 5%", y reglas como "SI el perfil es conservador Y el monto a invertir supera $50,000, ENTONCES recomendar bonos gubernamentales". Los *hechos* son declaraciones sobre el *estado del mundo*, mientras que las *reglas* son condicionales que describen cómo se relacionan los *hechos* entre sí.

🧠 **Motor de inferencia**: El *motor de inferencia* es el componente que aplica las *reglas* de la *base de conocimiento* para derivar conclusiones o tomar decisiones. Por ejemplo, en un sistema de diagnóstico médico, el motor de inferencia podría usar deducción ("SI el paciente tiene fiebre Y tos, ENTONCES puede tener gripe"), inducción (aprender patrones de casos previos para establecer nuevas reglas) y abducción (buscar la mejor explicación para un conjunto de síntomas) para llegar a un diagnóstico probable. Utiliza estas técnicas de razonamiento para procesar la información de manera similar a como lo haría un experto humano.

👥 **Interfaz de usuario**: Los sistemas expertos incluyen una *interfaz de usuario* que permite a los usuarios interactuar con el sistema. Por ejemplo, en un sistema experto legal, la interfaz podría presentar un cuestionario paso a paso donde el abogado ingresa los detalles del caso, y luego mostrar recomendaciones sobre precedentes legales relevantes y posibles estrategias de litigio. La interfaz debe ser intuitiva y fácil de usar, quizás incluyendo elementos visuales como diagramas de flujo o árboles de decisión para ayudar a los usuarios a entender el proceso.

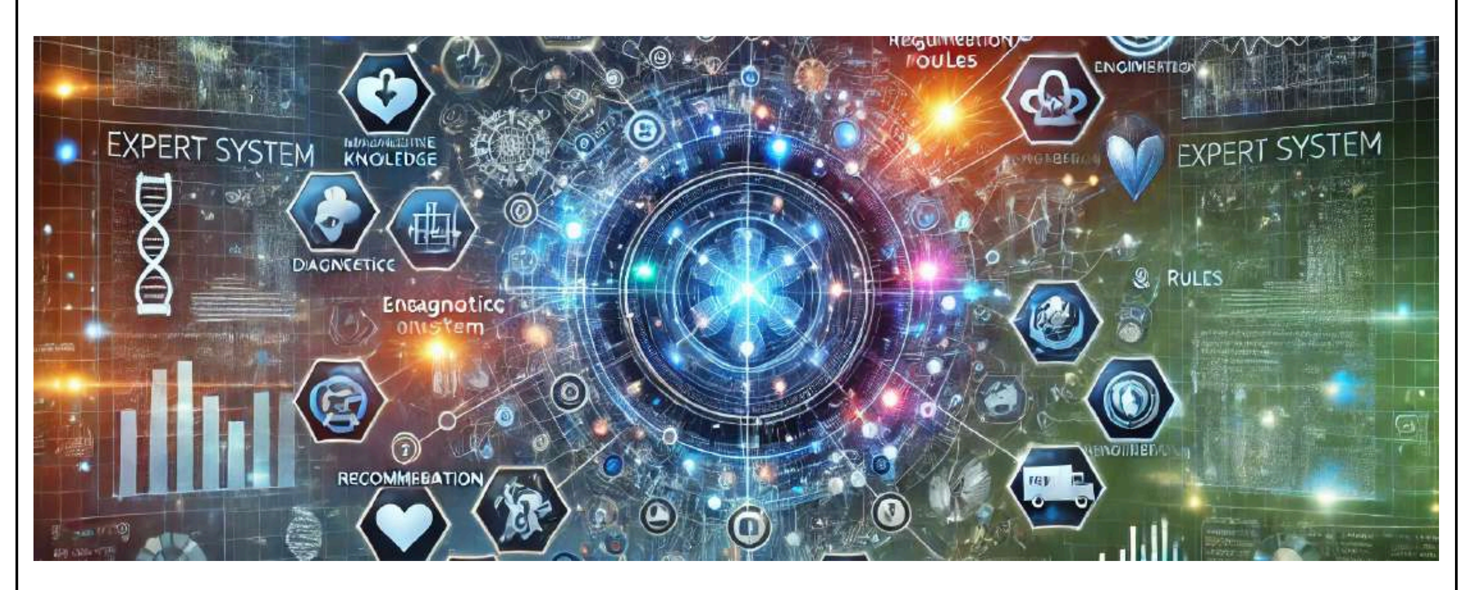
🔍 **Módulo de explicación**: Para aumentar la confianza y la aceptación del usuario, muchos sistemas expertos incluyen un *módulo de explicación*. Por ejemplo, en un sistema de aprobación de créditos, cuando rechaza una solicitud, el sistema no solo dice "NO", sino que explica: "La solicitud fue rechazada porque el ratio de deuda/ingreso del 45% supera nuestro límite del 40%, y el historial crediticio muestra 2 pagos atrasados en los últimos 6 meses". Este tipo de explicación ayuda a los usuarios a entender y confiar en las decisiones del sistema.

🧠 **Adquisición de conocimiento**: La adquisición de conocimiento es el proceso de capturar el conocimiento de expertos humanos y codificarlo en la *base de conocimiento* del sistema. Por ejemplo, para crear un sistema experto en mantenimiento de aeronaves, los desarrolladores podrían entrevistar a mecánicos experimentados, observar cómo diagnostican problemas, revisar manuales de mantenimiento y documentar

[80] Hoy, estamos viviendo un renacimiento de los sistemas expertos gracias a los Modelos de Lenguaje Grandes (LLM). Imaginá que los sistemas expertos tradicionales eran como libros de reglas muy específicas y rígidas, donde todo tenía que estar perfectamente definido de antemano. Ahora, con los LLM, estamos creando sistemas más flexibles y naturales.Cuando usamos fine-tuning, es como si "educáramos" al modelo para que se vuelva experto en un tema específico, por ejemplo medicina o leyes. Es similar a cuando entrenas a un profesional, pero en este caso entrenas al modelo con ejemplos. Y el RAG es como darle al modelo una biblioteca de consulta en tiempo real. En vez de tener que memorizar todo, el modelo puede "buscar" información actualizada cuando la necesita, igual que un médico consultando artículos recientes para mantenerse al día. La gran diferencia es que estos nuevos sistemas pueden entender y responder en lenguaje natural - como si estuvieras hablando con un experto real - en vez de seguir reglas rígidas y predefinidas como hacían los sistemas antiguos.

procedimientos paso a paso. Este proceso puede llevar meses o años, ya que implica no solo recopilar información, sino también estructurarla de manera que el sistema pueda utilizarla efectivamente.

Tipos de Sistemas Expertos



📏 **Sistemas basados en reglas**: Utilizan reglas *if-then (si-entonces)* para representar el conocimiento, aplicando los principios que ya estudiamos en las secciones de lógica proposicional y lógica de primer orden. Cada regla describe una relación entre una condición y una acción o conclusión, siguiendo la misma estructura de implicación lógica ($P \rightarrow Q$) que analizamos anteriormente. Por ejemplo, en un sistema experto médico para diagnóstico de gripe, una regla podría ser: "SI el paciente tiene fiebre mayor a 38°C Y tiene dolor muscular Y tiene congestión nasal, ENTONCES existe una probabilidad del 80% de que tenga gripe". Esta regla utiliza los operadores lógicos AND ($\land$) que ya conocemos, demostrando cómo los fundamentos de la lógica se aplican directamente en sistemas prácticos. Estos sistemas son fáciles de entender y modificar, pero pueden volverse complejos y difíciles de mantener a medida que crecen, especialmente cuando se necesitan múltiples reglas interconectadas.

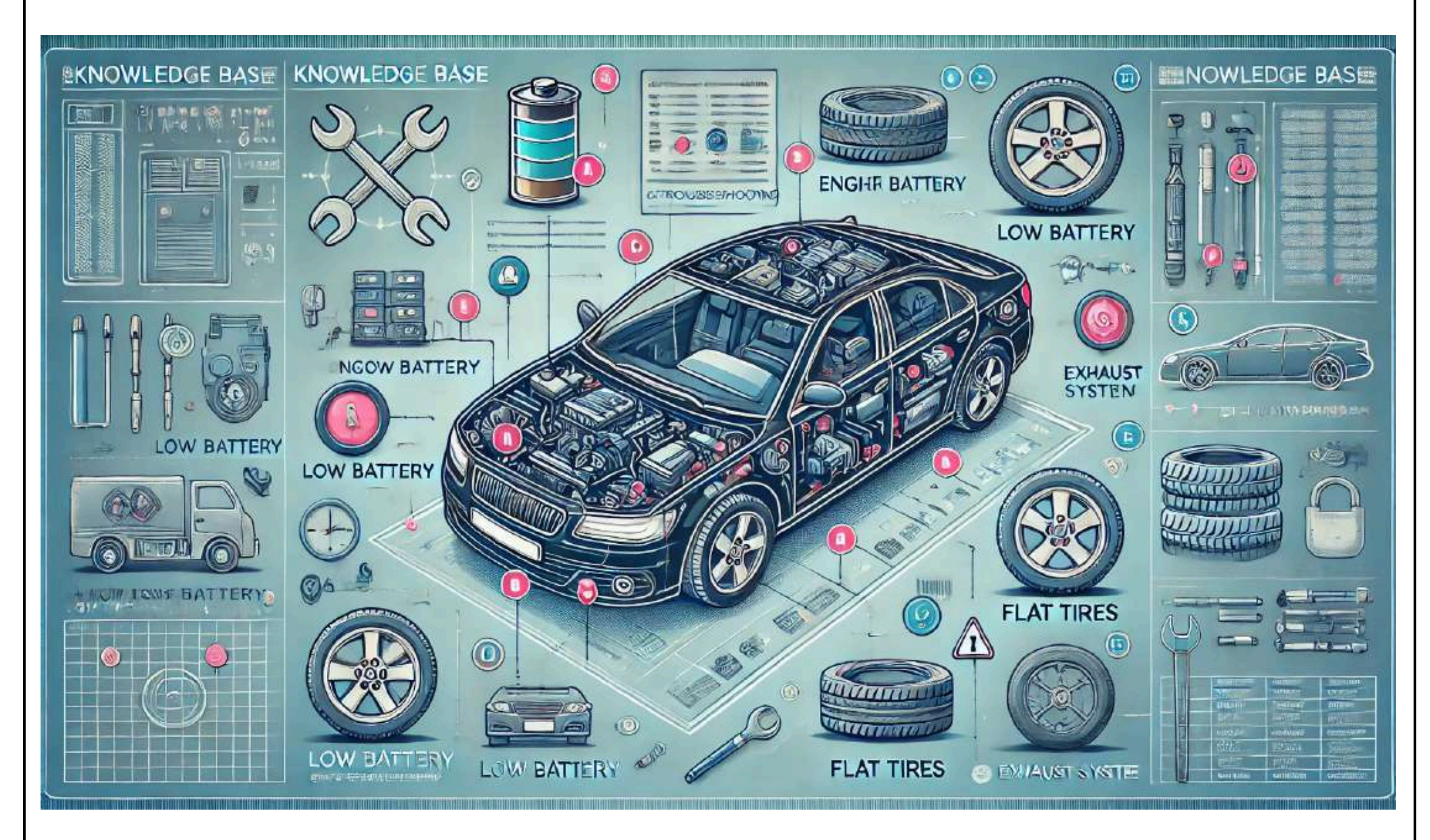
📚 **Sistemas basados en casos (*Case-Based Reasoning*)**: Resuelven nuevos problemas buscando soluciones a problemas similares en una base de datos de casos previos. Por ejemplo, un sistema de soporte técnico podría almacenar miles de problemas resueltos anteriormente. Cuando un usuario reporta que su impresora no funciona y describe los síntomas, el sistema busca casos similares en su base de datos y sugiere soluciones que funcionaron en situaciones parecidas, como "verificar conexión USB" o "reinstalar drivers". Este enfoque es útil en dominios donde los problemas no pueden ser fácilmente descritos por reglas fijas.

🌐 **Sistemas basados en Redes Bayesianas**: Utilizan redes probabilísticas para representar y razonar sobre la incertidumbre en el conocimiento. Por ejemplo, en un sistema de diagnóstico automotriz, una red bayesiana podría relacionar síntomas como "el motor hace ruidos extraños" con posibles causas como "falla en la bomba de aceite" o "problemas en los rodamientos", asignando probabilidades basadas en la evidencia observada. Si el ruido es metálico y proviene de la parte inferior del motor, el sistema podría calcular que hay un 70% de probabilidad de que sea un problema en los rodamientos. Estos sistemas son efectivos para manejar situaciones donde la información es incompleta o incierta. De hecho en la siguiente sección veremos este tema detalladamente.

🖼️ **Sistemas basados en marcos (*Frame-Based Systems*)**: Utilizan estructuras de datos llamadas *marcos* para representar objetos y sus relaciones. Por ejemplo, en un sistema de configuración de computadoras, un marco "Computadora" podría contener atributos como "procesador", "memoria RAM", "almacenamiento" y

"tarjeta gráfica", junto con reglas de compatibilidad entre componentes. Cuando un usuario selecciona un procesador específico, el sistema automáticamente actualiza las opciones disponibles para los demás componentes basándose en las relaciones definidas en los marcos. Los *marcos* pueden contener atributos, valores y procedimientos para manipular la información.

Ejemplo de un sistema experto basado en reglas



Base de conocimientos: diagnóstico de problemas en automóviles
Esta base de conocimientos representa un sistema experto para el diagnóstico de problemas automotrices, diseñado para ayudar en la identificación y resolución de problemas comunes en vehículos.

Está estructurada en cuatro componentes principales:

- Una tabla de **síntomas** que cataloga las manifestaciones observables de problemas (como el auto que no arranca o ruidos extraños),
- Una tabla de **estados del vehículo** que define las mediciones técnicas específicas (como niveles de batería y aceite),
- Una tabla de **reglas de diagnóstico** que establece las relaciones lógicas entre síntomas y problemas potenciales (usando condiciones AND para mayor precisión), y
- Una tabla de **niveles de urgencia y acciones** que proporciona recomendaciones claras sobre cómo proceder según la gravedad del problema.

Este sistema permite un enfoque sistemático para el diagnóstico automotriz, donde cada problema se evalúa con un nivel de confianza específico y se proporcionan tanto acciones inmediatas como preventivas, facilitando la toma de decisiones informada tanto para mecánicos como para usuarios finales.

Síntomas

ID	Síntoma	Tipo	Pregunta	Valores Posibles
S001	No arranca	Binario	¿El automóvil no arranca cuando gira la llave?	sí/no
S002	Luces no encienden	Binario	¿Las luces del automóvil no encienden?	sí/no
S003	Ruidos motor	Categórico	¿Qué tipo de ruido hace el motor?	golpeteo/chirrido/traqueteo/ninguno
S004	Humo motor	Categórico	¿De qué color es el humo que sale del motor?	azul/blanco/negro/ninguno
S005	Consumo combustible	Binario	¿Ha notado mayor consumo de combustible?	sí/no

Estados del Vehículo

ID	Estado	Tipo	Unidad/Valores	Rango Normal
E001	Nivel batería	Numérico	Voltios	12.0 - 14.0 V
E002	Nivel aceite	Categórico	bajo/normal/alto	normal
E003	Nivel refrigerante	Categórico	bajo/normal/alto	normal

Reglas de Diagnóstico

ID	Condiciones	Problema	Confianza	Recomendación Principal
R001	S001=sí AND S002=sí	Batería descargada	90%	Verificar conexiones y cargar/reemplazar batería
R002	S001=sí AND S002=no	Motor de arranque	85%	Revisar motor de arranque
R003	S003=golpeteo AND S009=sí	Falta de aceite	95%	Verificar nivel de aceite inmediatamente

Niveles de Urgencia y Acciones

Problema	Urgencia	Acciones Inmediatas	Acciones Preventivas
Batería descargada	Media	Usar cables pasa corriente	Revisar sistema de carga
Falta de aceite	Alta	Detener el vehículo	Programar cambios de aceite regulares
Fuga de combustible	Crítica	Estacionar en área ventilada, no arrancar	Inspección regular de líneas de combustible

Hechos

Cuando un usuario reporta un problema, los hechos se convierten en entradas para nuestro sistema experto.

Por ejemplo:

1. El automóvil no arranca
2. Las luces no encienden
3. El motor hace ruidos extraños
4. Hay humo saliendo del motor
5. El automóvil consume más combustible de lo normal
6. Los frenos hacen ruido al usarlos
7. El volante vibra al conducir
8. El aire acondicionado no enfría

9. Hay un olor a quemado
10. El automóvil se sobrecalienta

Reglas

Las reglas representan la lógica de diagnóstico que conecta los síntomas observados (hechos) con posibles problemas y sus soluciones. Estas reglas típicamente siguen una estructura SI-ENTONCES (IF-THEN), donde "SI" describe una combinación de síntomas observados, y "ENTONCES" proporciona el diagnóstico probable junto con su nivel de confianza y las acciones recomendadas. Por ejemplo:

1. SI el automóvil no arranca Y las luces no encienden ENTONCES el problema probable es la batería
2. SI el automóvil no arranca Y las luces encienden ENTONCES el problema probable es el motor de arranque
3. SI el motor hace ruidos extraños Y hay un olor a quemado ENTONCES el problema probable es falta de aceite
4. SI hay humo saliendo del motor Y el humo es azul ENTONCES el problema probable es aceite quemándose
5. SI hay humo saliendo del motor Y el humo es blanco ENTONCES el problema probable es fuga de refrigerante
6. SI el automóvil consume más combustible de lo normal Y hay un olor a gasolina ENTONCES el problema probable es una fuga de combustible
7. SI los frenos hacen ruido al usarlos Y el ruido es un chirrido agudo ENTONCES el problema probable es desgaste de las pastillas de freno
8. SI el volante vibra al conducir Y la vibración aumenta con la velocidad ENTONCES el problema probable es un desbalanceo de las ruedas
9. SI el aire acondicionado no enfría Y se escucha el compresor funcionando ENTONCES el problema probable es falta de refrigerante
10. SI el automóvil se sobrecalienta Y el nivel de refrigerante es bajo ENTONCES el problema probable es una fuga en el sistema de refrigeración

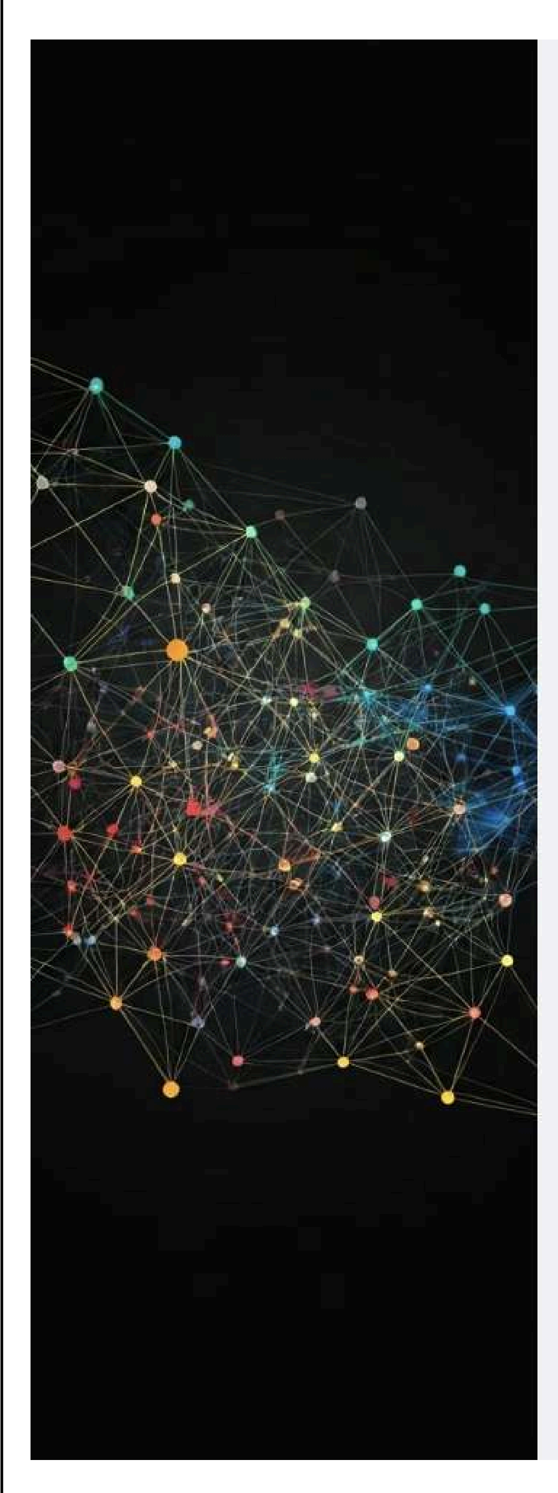
Recomendaciones

Las recomendaciones en un sistema experto automotriz representan las acciones específicas que deben tomarse una vez identificado un problema, clasificadas según su urgencia y divididas en acciones inmediatas y preventivas. Cada recomendación está directamente vinculada a un diagnóstico específico y está diseñada para guiar al usuario sobre cómo proceder de manera segura y efectiva, desde soluciones temporales hasta reparaciones permanentes. Por ejemplo:

1. Problema: *Batería*. **Recomendación**: Verificar las conexiones de la batería y cargarla. Si es antigua, considerar reemplazarla.
2. Problema: *Motor de arranque*. **Recomendación**: Llevar el vehículo a un mecánico para verificar y posiblemente reemplazar el motor de arranque.
3. Problema: *Falta de aceite*. **Recomendación**: Verificar el nivel de aceite inmediatamente. Agregar aceite si es necesario y programar un cambio de aceite.
4. Problema: *Aceite quemándose*. **Recomendación**: Llevar el vehículo a un mecánico para verificar posibles fugas o problemas en los anillos del pistón.
5. Problema: *Fuga de refrigerante*. **Recomendación**: Verificar el nivel de refrigerante y buscar fugas visibles. Llevar a un mecánico para una inspección detallada.
6. Problema: *Fuga de combustible*. **Recomendación**: No conducir el vehículo. Hacer remolcar a un taller mecánico para una inspección y reparación inmediata.
7. Problema: *Desgaste de las pastillas de freno*. **Recomendación**: Programar una revisión y posible reemplazo de las pastillas de freno.
8. Problema: *Desbalanceo de las ruedas*. **Recomendación**: Llevar el vehículo a un taller para balancear

las ruedas.

9. Problema: *Falta de refrigerante en el aire acondicionado*. **Recomendación**: Llevar a un especialista en aire acondicionado para recargar el sistema y verificar posibles fugas.
10. Problema: *Fuga en el sistema de refrigeración*. **Recomendación**: No conducir el vehículo cuando está sobrecalentado. Hacer remolcar a un mecánico para reparar la fuga.



Tipos de Sistemas Expertos

1 Basados en Reglas

Utilizan reglas *if-then* para representar el conocimiento, como un sistema de diagnóstico médico que determine el tratamiento adecuado basado en los síntomas del paciente.

2 Basados en Casos

Resuelven problemas buscando soluciones a casos similares previos, como un sistema que recomiende productos a los clientes basado en sus compras anteriores.

3 Redes Bayesianas

Usan redes probabilísticas para razonar sobre la incertidumbre, como un sistema que evalúe el riesgo de una inversión financiera considerando múltiples factores.

4 Basados en Marcos

Emplean estructuras de datos para representar objetos y relaciones, como un sistema que clasifique tipos de vinos basado en características como la variedad de uva, la región de origen y el año de cosecha.

Sistemas Expertos vs Árboles de Decisión (Machine Learning)

Te podrás preguntar qué diferencia existe entre un sistema experto y un árbol de decisión, después de todo,

ambos parecen seguir una estructura similar de "si esto, entonces aquello" y ambos nos ayudan a tomar decisiones basadas en ciertas condiciones. Esta es una pregunta muy válida, especialmente considerando que en la superficie, tanto los sistemas expertos como los árboles de decisión parecen resolver problemas de manera similar, evaluando condiciones y llegando a conclusiones.

La principal diferencia radica en su origen y forma de construcción. Los sistemas expertos son como tener un experto humano digitalizado: se construyen cuidadosamente capturando el conocimiento específico y la experiencia de profesionales en el campo. Imagina a un equipo de ingenieros entrevistando durante meses a médicos experimentados para entender exactamente cómo diagnostican enfermedades, traduciendo ese conocimiento en reglas precisas y estructuradas. Este conocimiento es explícito, se puede revisar y modificar directamente, y lo más importante, puede explicar su razonamiento de manera que otros expertos lo entiendan, justificando cada paso de su proceso de decisión.

En contraste, los árboles de decisión son como aprendices que estudian miles de ejemplos históricos para descubrir patrones por sí mismos. En lugar de recibir reglas explícitas de expertos, estos sistemas utilizan algoritmos matemáticos para analizar grandes cantidades de datos y detectar automáticamente patrones y relaciones. Por ejemplo, podrían analizar miles de registros médicos históricos para aprender qué combinaciones de síntomas suelen indicar ciertas enfermedades. Aunque pueden ser muy precisos y eficientes, especialmente con grandes volúmenes de datos, pueden perder algunos de los matices y la intuición que un experto humano (y por ende, un sistema experto) manejaría naturalmente. Es como la diferencia entre aprender medicina estudiando miles de casos versus aprender directamente de un médico experimentado que puede explicar el "por qué" detrás de cada decisión.

La verdadera belleza de la inteligencia artificial radica precisamente en esta diversidad de enfoques para resolver problemas similares. Mientras los sistemas expertos capturan la sabiduría humana acumulada a través de años de experiencia, traduciéndola en reglas explícitas y razonamiento estructurado, los árboles de decisión y otros métodos de machine learning descubren sus propias reglas a través del análisis de datos masivos. No se trata de determinar qué enfoque es mejor, sino de reconocer que cada uno tiene sus propias fortalezas y aplicaciones ideales. Esta versatilidad de la IA nos permite elegir el método más apropiado según el contexto, los recursos disponibles y los objetivos específicos, o incluso combinar múltiples enfoques para crear soluciones híbridas más robustas. Es esta flexibilidad y multiplicidad de caminos lo que hace que la IA sea una herramienta tan poderosa y fascinante en nuestra búsqueda por replicar y potenciar las capacidades del razonamiento humano.

Sistemas Expertos vs LLMs (Large Language Models)

Al igual que con los árboles de decisión, es natural preguntarse cuál es la diferencia entre los sistemas expertos tradicionales y los modelos de lenguaje grande (LLMs) modernos, especialmente considerando que ambos pueden proporcionar respuestas expertas y recomendaciones en diversos campos. Sin embargo, existen diferencias fundamentales en su diseño, funcionamiento y capacidades.

Los sistemas expertos son como una biblioteca altamente organizada y especializada: se construyen con reglas explícitas y conocimiento estructurado en un dominio específico. Por ejemplo, un sistema experto médico contiene reglas precisas sobre síntomas y diagnósticos, codificadas manualmente por expertos en el campo. Su razonamiento es determinista y transparente, siguiendo siempre las mismas reglas lógicas para llegar a conclusiones específicas. Cuando un sistema experto dice que un paciente probablemente tiene gripe, puede explicar exactamente qué reglas siguió para llegar a esa conclusión.

En contraste, los LLMs son como un estudiante que “leyó” prácticamente todo lo disponible en internet: son entrenados con cantidades masivas de texto que abarcan múltiples dominios del conocimiento. En lugar de seguir reglas explícitas, utilizan patrones estadísticos complejos aprendidos de sus datos de entrenamiento para generar respuestas. Pueden abordar una gama mucho más amplia de temas y son más flexibles en su capacidad para entender y responder preguntas en lenguaje natural, pero sus respuestas pueden ser menos

predecibles y más difíciles de verificar.

Algunas diferencias clave:

1. **Especialización vs generalización**: Los sistemas expertos pueden ser justamente expertos en dominios muy específicos, mientras que los LLMs son generalistas con conocimiento amplio pero potencialmente menos profundo en áreas específicas.
2. **Transparencia vs caja negra**: Los sistemas expertos tienen un razonamiento transparente y trazable, mientras que los LLMs funcionan como una "caja negra" donde es más difícil entender exactamente cómo llegaron a una conclusión específica.
3. **Actualización del conocimiento**: Los sistemas expertos se pueden actualizar modificando directamente sus reglas y base de conocimiento, mientras que los LLMs requieren reentrenamiento o fine-tuning para incorporar nuevo conocimiento.
4. **Interacción**: Los sistemas expertos típicamente siguen un formato de preguntas y respuestas estructurado, mientras que los LLMs pueden mantener conversaciones más naturales y contextuales.
5. **Recursos computacionales**: Los sistemas expertos son generalmente más ligeros y requieren menos recursos computacionales, mientras que los LLMs necesitan significativa potencia de procesamiento y memoria.

La combinación de ambos enfoques está emergiendo como una tendencia prometedora. Por ejemplo, algunos sistemas modernos usan LLMs para la interfaz de usuario y el procesamiento de lenguaje natural, mientras mantienen sistemas expertos en el backend para tareas que requieren precisión y confiabilidad absoluta, como el diagnóstico médico o el control de calidad en manufactura. Esta sinergia permite aprovechar la flexibilidad y naturalidad de los LLMs junto con la precisión y confiabilidad de los sistemas expertos tradicionales.

Aplicaciones de los Sistemas Expertos

🏥 **Medicina**: Los sistemas expertos médicos pueden ayudar a los médicos a diagnosticar enfermedades, recomendar tratamientos y prever posibles complicaciones. Ejemplos incluyen sistemas de diagnóstico como MYCIN y sistemas de planificación de tratamientos como ONCOCIN.

🛠️ **Ingeniería**: En el campo de la ingeniería, los sistemas expertos se utilizan para diseñar componentes complejos, analizar fallos y optimizar procesos de fabricación. Pueden ayudar a los ingenieros a tomar decisiones informadas y mejorar la eficiencia.

💰 **Finanzas**: Los sistemas expertos financieros pueden proporcionar asesoría en inversiones, gestionar carteras y detectar fraudes. Utilizan reglas y modelos para analizar datos financieros y tomar decisiones precisas y oportunas.

🌾 **Agricultura**: En la agricultura, los sistemas expertos ayudan a los agricultores a gestionar cultivos, optimizar el uso de recursos y predecir rendimientos. Pueden proporcionar recomendaciones sobre el mejor momento para sembrar y cosechar, y cómo manejar plagas y enfermedades.

⚖️ **Derecho**: Los sistemas expertos legales pueden asistir a abogados y jueces en la investigación de casos, la interpretación de leyes y la preparación de documentos legales. Ayudan a garantizar que se sigan procedimientos legales correctos y a reducir errores.

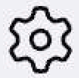
🎧 **Servicio al Cliente**: En el servicio al cliente, los sistemas expertos pueden automatizar la resolución de problemas comunes, proporcionar respuestas rápidas a las consultas y guiar a los clientes a través de procedimientos de solución de problemas.

Aplicaciones de los Sistemas Expertos



Medicina

Ayudan en diagnósticos y recomendaciones de tratamientos, como un sistema que analiza síntomas y datos de pacientes para sugerir el mejor plan de tratamiento para una enfermedad específica.



Ingeniería

Diseñan componentes y optimizan procesos de fabricación, como un sistema que ayuda a diseñar el layout óptimo de una planta de producción para maximizar la eficiencia.



Finanzas

Proporcionan asesoría en inversiones y detectan fraudes, como un sistema que analiza patrones de transacciones para identificar actividades sospechosas de fraude.

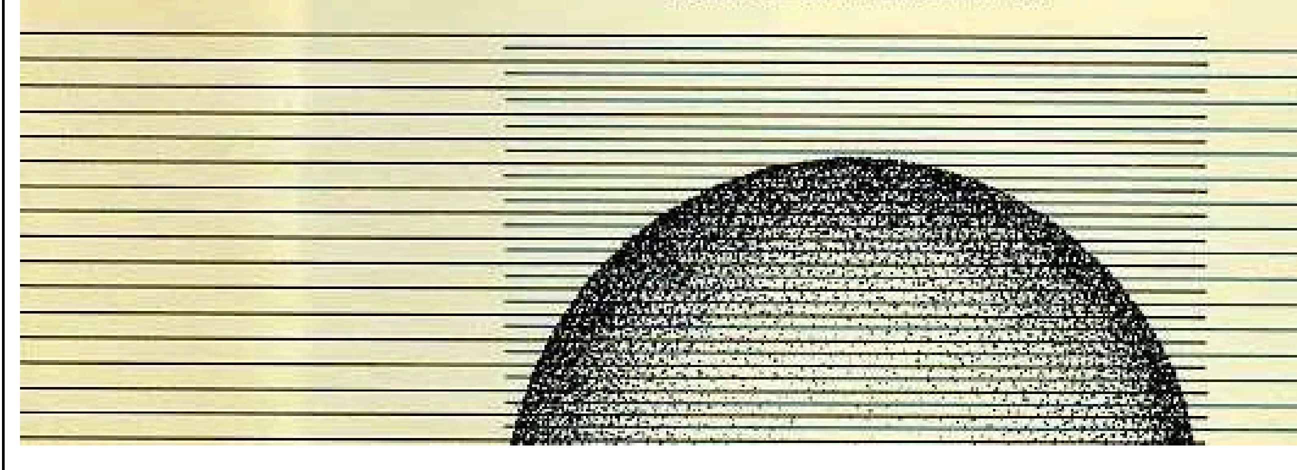


Derecho

Asisten en investigación de casos e interpretación de leyes, como un sistema que ayuda a los abogados a buscar precedentes legales relevantes para un caso específico.

Algunos de los sistemas expertos más famosos y emblemáticos en la historia de la inteligencia artificial son:

1. **MYCIN (1970s):**
   - **Descripción**: Uno de los primeros sistemas expertos en medicina. Fue desarrollado en la Universidad de Stanford para diagnosticar infecciones bacterianas y recomendar tratamientos basados en antibióticos.
   - Impacto: Pionero en el uso de reglas heurísticas y el razonamiento basado en conocimiento **experto**.



2. **DENDRAL (1960s-1970s):**
   - **Descripción**: Otro sistema desarrollado en Stanford, enfocado en la química orgánica. Ayudaba a los científicos a identificar estructuras moleculares desconocidas utilizando datos espectrométricos.
   - **Impacto**: Considerado uno de los primeros programas de IA diseñado para abordar problemas científicos específicos.
3. **XCON (también conocido como R1) (1980s):**
   - **Descripción**: Desarrollado por Digital Equipment Corporation (DEC) para configurar sistemas informáticos complejos. Utilizaba reglas para seleccionar componentes de hardware basados en las especificaciones del cliente.
   - **Impacto**: Ahorro de costos significativos para DEC al automatizar la configuración de sistemas, siendo un éxito comercial importante.
4. **CLIPS (1985):**
   - **Descripción**: Un lenguaje de programación desarrollado por NASA para construir sistemas expertos. Es ampliamente utilizado debido a su flexibilidad y facilidad para codificar reglas basadas en conocimiento.
   - **Impacto**: Adoptado en diversos campos, desde la NASA hasta la industria privada, debido a su robustez.
5. **Prolog (1970s):**
   - **Descripción**: Aunque es más un lenguaje de programación lógico que un sistema experto en sí, Prolog ha sido fundamental en el desarrollo de sistemas expertos y la resolución de problemas mediante reglas lógicas.
   - **Impacto**: Utilizado ampliamente en la investigación de IA y en la creación de sistemas expertos.

6. **CYC (1984):**
   - **Descripción**: Proyecto ambicioso que intenta recopilar conocimiento común del mundo en una base de datos enorme, para que las máquinas puedan "razonar" como lo haría un ser humano.
   - **Impacto**: Aunque sigue en desarrollo, ha sido influyente en la creación de sistemas de razonamiento basados en grandes bases de conocimiento.
7. **Expert Systems Shells (Herramientas para construir sistemas expertos):**
   - **Ejemplos**: Herramientas como EMYCIN, derivado del sistema MYCIN, y OPS5, que permitieron a los desarrolladores crear sus propios sistemas expertos sin tener que empezar desde cero.
   - **Impacto**: Facilitaron la creación y adopción de sistemas expertos en muchas industrias.
8. **IBM Watson (2011-presente):**
   - **Descripción**: Sistema de inteligencia artificial desarrollado por IBM, capaz de responder preguntas formuladas en lenguaje natural. Ganó fama al vencer a campeones humanos en el programa de concursos Jeopardy! en 2011.



   - **Impacto**: Revolucionó el procesamiento del lenguaje natural y la aplicación de IA en diversos campos como medicina, finanzas y atención al cliente. Demuestra la capacidad de los sistemas expertos modernos para procesar grandes cantidades de datos no estructurados y generar respuestas complejas.
   - **Consideraciones y matices:**
      i. Características híbridas: Watson combina elementos de sistemas expertos tradicionales con tecnologías de IA más modernas como el *deep learning* y el procesamiento del lenguaje natural.
      ii. Evolución del sistema: Aunque Watson comenzó con características más cercanas a los sistemas expertos tradicionales, ha evolucionado significativamente desde su aparición en Jeopardy!
      iii. Enfoque basado en datos: Gran parte del poder de Watson proviene de su capacidad para procesar y analizar grandes cantidades de datos no estructurados, lo que lo diferencia de los sistemas expertos clásicos basados principalmente en reglas.
      iv. Variedad de aplicaciones: Watson se ha adaptado a diversos campos, desde medicina hasta finanzas, lo que lo hace más versátil que un sistema experto tradicional enfocado en un dominio específico.
      v. Aprendizaje continuo: A diferencia de los sistemas expertos clásicos, Watson tiene

capacidades de aprendizaje automático que le permiten mejorar con el tiempo.

Con esto damos por finalizada nuestra sección sobre *Sistemas Expertos* y damos paso a las ya mencionadas *Redes Bayesianas*. Éstas representan una evolución natural de los Sistemas Expertos porque introducen un marco probabilístico más sofisticado que puede manejar la incertidumbre. Esto es particularmente relevante dado que el mundo real rara vez opera en términos absolutos de verdadero o falso, sino más bien en grados de probabilidad e incertidumbre. Mientras los Sistemas Expertos tradicionales se basan en reglas deterministas del tipo SI-ENTONCES, las Redes Bayesianas utilizan probabilidades condicionales para representar las relaciones entre variables. Esta transición fue crucial en la inteligencia artificial dado que las Redes Bayesianas pueden atrapar relaciones más complejas y sutiles entre variables, adaptarse a nueva evidencia y proporcionar inferencias probabilísticas más precisas. En el ejemplo del diagnóstico de automóviles, una Red Bayesiana podría asignar probabilidades a diferentes causas del fallo basándose en los múltiples síntomas observados, considerando no solo la presencia o ausencia de un síntoma, sino también su intensidad y su relación probabilística con otros síntomas y causas.

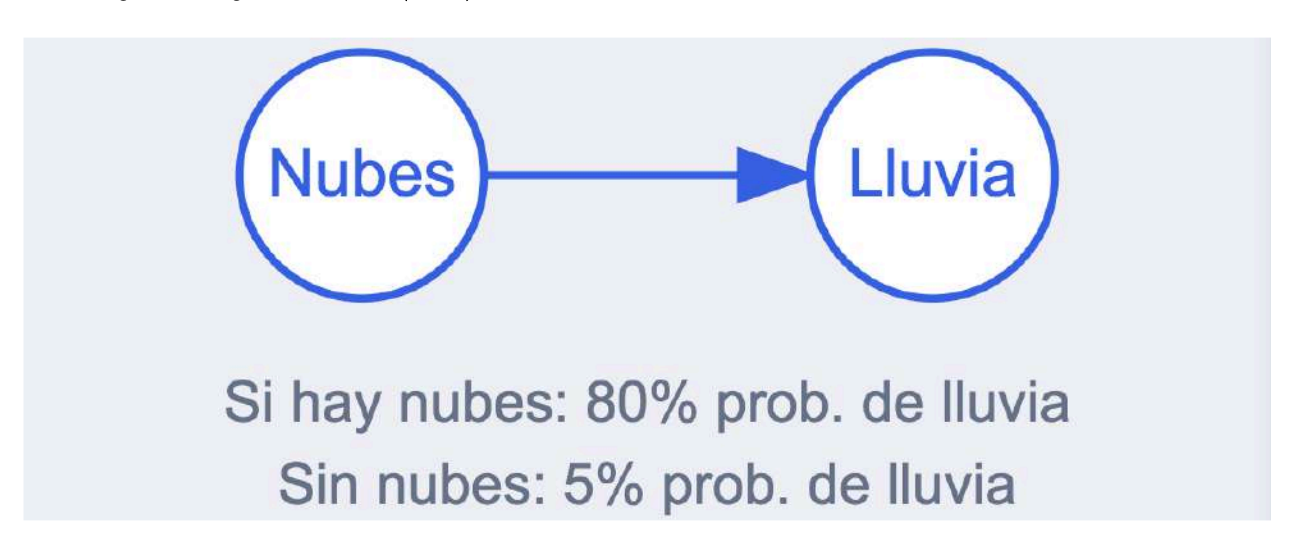
10. 🔗 Redes Bayesianas



Redes Bayesianas: Modelado Probabilístico

Las redes bayesianas son poderosas herramientas de modelado probabilístico que representan de manera gráfica las relaciones de dependencia entre variables. Estas redes son esenciales para la toma de decisiones en entornos inciertos, ya que permiten cuantificar y analizar la incertidumbre inherente a diferentes fenómenos.

Las redes bayesianas son herramientas matemáticas que nos permiten modelar situaciones donde diferentes eventos están conectados mediante relaciones de causa y efecto, usando probabilidades para describir qué tan fuerte es cada conexión. En esencia, son grafos donde cada punto (o nodo) representa una variable o evento, y las flechas entre estos puntos muestran cómo se influyen entre sí. Imaginá que tenés un árbol genealógico: así como los padres influyen en las características de sus hijos, en una red bayesiana, unas variables influyen en otras siguiendo un orden lógico y sin crear círculos cerrados - esto es lo que llamamos un *grafo dirigido acíclico* (DAG).



También conocidas como *redes de creencias*, las redes bayesianas combinan dos conceptos importantes: la **teoría de probabilidad** (que nos ayuda a calcular qué tan probable es que algo suceda) y la **teoría de grafos** (que nos permite representar conexiones entre elementos mediante dibujos con puntos y flechas). Es como tener un mapa del clima donde podemos ver cómo la temperatura, la humedad y la presión atmosférica se relacionan entre sí para predecir si lloverá.

Estas redes son herramientas muy versátiles y poderosas para modelar situaciones donde hay incertidumbre. Por ejemplo:

- En **medicina**, ayudan a los doctores a diagnosticar enfermedades basándose en los síntomas del paciente, superando las limitaciones de los sistemas expertos tradicionales y la lógica de primer orden. Mientras que un sistema experto utilizaría reglas rígidas del tipo "SI tiene estos síntomas ENTONCES tiene esta enfermedad", y la lógica de primer orden necesitaría predicados exactos como "TieneFiebre(paciente)", las redes bayesianas manejan la incertidumbre de forma natural. Por ejemplo, si un paciente presenta fiebre, dolor de garganta y fatiga, la red puede calcular la probabilidad de que tenga gripe, COVID-19 u otra enfermedad, considerando también factores como la edad del paciente y la época del año. Esta capacidad de manejar probabilidades y relaciones inciertas las hace especialmente valiosas en medicina, donde rara vez hay diagnósticos 100% seguros y donde múltiples factores pueden influir en la probabilidad de una enfermedad.
- En **ingeniería**, pueden predecir cuándo una máquina podría fallar basándose en distintas señales de advertencia. Por ejemplo, en una turbina eólica, la red puede analizar datos como vibraciones anormales (si las vibraciones superan los 7mm/s), temperatura (cuando excede los 85°C en los rodamientos), y ruidos inusuales (frecuencias superiores a 100Hz) para calcular la probabilidad de una falla. Si la red detecta vibraciones de 8mm/s junto con una temperatura de 90°C, podría determinar que hay un 75% de probabilidad de que la turbina necesite mantenimiento en los próximos 30 días, permitiendo programar reparaciones preventivas antes de una falla crítica.
- En **análisis de riesgos**, ayudan a empresas y organizaciones a tomar mejores decisiones evaluando diferentes escenarios posibles. Por ejemplo, un banco puede usar una red bayesiana para evaluar el riesgo de un préstamo, considerando factores como el historial crediticio del cliente (si tiene más de 2 pagos atrasados en los últimos 6 meses), sus ingresos (si son 3 veces mayores que la cuota mensual), su empleo actual (si tiene más de 2 años de antigüedad) y las condiciones económicas del mercado (si la tasa de desempleo es superior al 8%). La red podría determinar que un cliente con un atraso en pagos, ingresos 4 veces mayores a la cuota, 3 años en su trabajo actual, pero en un mercado con 10% de desempleo, tiene un 65% de probabilidad de pagar el préstamo sin problemas, un 25% de probabilidad de tener algunos retrasos y un 10% de riesgo de default.

La belleza de las redes bayesianas además es que pueden aprender de la experiencia y actualizar sus predicciones a medida que reciben nueva información, igual que nosotros aprendemos de nuestras experiencias pasadas para tomar mejores decisiones en el futuro.

A diferencia de los sistemas expertos que vimos anteriormente, que se basan en reglas rígidas del tipo "si-entonces" definidas por expertos humanos, las redes bayesianas ofrecen un enfoque más flexible y realista para manejar la incertidumbre. Mientras que un sistema experto podría decir "si el paciente tiene fiebre Y tos, ENTONCES tiene gripe" de manera determinista, una red bayesiana expresaría esto en términos de probabilidades, reconociendo que estos síntomas podrían indicar gripe con, por ejemplo, un 70% de probabilidad, pero también podrían sugerir un resfriado común (20%) u otra enfermedad (10%). Esta capacidad para manejar múltiples posibilidades simultáneamente y actualizar estas probabilidades a medida que se obtiene nueva información hace que las redes bayesianas sean especialmente útiles en situaciones del mundo real, donde rara vez tenemos certeza absoluta y donde diferentes factores pueden influirse mutuamente de formas complejas.

Las redes bayesianas se consideran una forma de inteligencia artificial porque imitan uno de los aspectos

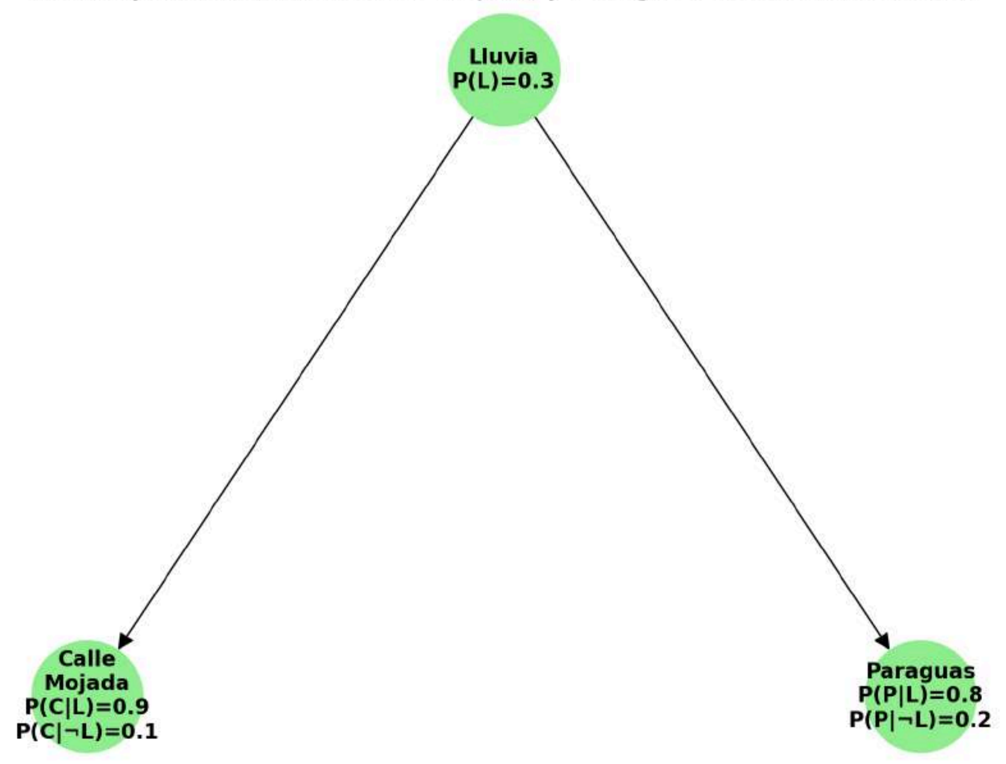
más fundamentales del razonamiento humano: **nuestra capacidad para aprender de la experiencia y tomar decisiones bajo incertidumbre**. Así como los humanos constantemente actualizamos nuestras creencias basándonos en nueva información (por ejemplo, si vemos nubes oscuras, aumenta nuestra creencia de que lloverá), las redes bayesianas pueden actualizar sus probabilidades a medida que reciben nuevos datos. Además, estas redes pueden "pensar" de manera similar a un experto humano, considerando múltiples factores simultáneamente y sus interrelaciones para llegar a conclusiones. Por ejemplo, un médico experimentado no solo considera cada síntoma de manera aislada, sino que evalúa cómo estos síntomas se relacionan entre sí y cómo ciertos factores (como la edad del paciente o sus antecedentes médicos) influyen en la probabilidad de diferentes diagnósticos - exactamente el mismo tipo de razonamiento que implementa una red bayesiana. Esta capacidad para emular el proceso de razonamiento y aprendizaje humano es lo que las convierte en una verdadera forma de inteligencia artificial.

Fundamentos de las Redes Bayesianas

Nodos y Aristas: Las redes bayesianas consisten en **nodos y aristas**. Cada **nodo** en el grafo representa una *variable aleatoria* que puede tomar uno de varios estados, y las **aristas** indican relaciones de *dependencia condicional* entre estas variables.

- **Nodos**: Representan *variables aleatorias*. Cada nodo tiene un conjunto de posibles valores (estados) y una distribución de probabilidad asociada.
- **Aristas**: Representan *dependencias condicionales* entre variables. Una arista de un nodo A a un nodo B implica que B depende condicionalmente de A.

Veamos un ejemplo:



En esta red bayesiana, cada nodo representa un evento, y las probabilidades asociadas se leen de la siguiente manera:

- **P(L)** representa la probabilidad de que ocurra el evento "Lluvia", que en este caso es 0.3 (30%).
- **P(C|L)** es la probabilidad condicional de que las calles estén mojadas dado que está lloviendo, que es 0.9 (90%).
- **P(C|¬L)** indica la probabilidad de que las calles estén mojadas cuando no está lloviendo, que es 0.1 (10%).
- De manera similar, **P(P|L)** es la probabilidad de que alguien lleve un paraguas si está lloviendo, 0.8 (80%), mientras que **P(P|¬L)** es la probabilidad de que alguien lleve un paraguas si no llueve, que es 0.2 (20%).

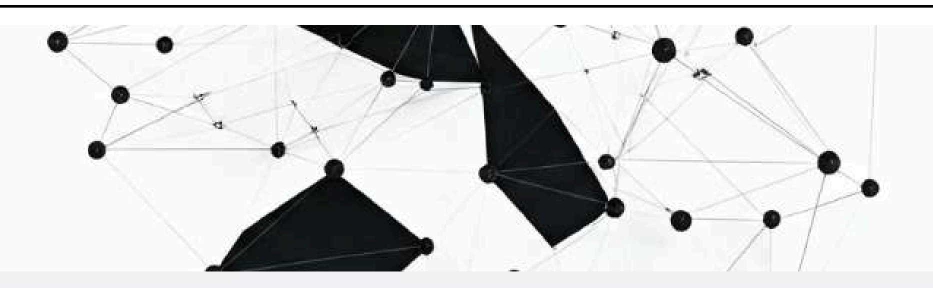
Estas probabilidades condicionales muestran cómo cada evento depende de la ocurrencia o no de "Lluvia".

Distribución de Probabilidad Conjunta: La Distribución de Probabilidad Conjunta en una red bayesiana es como un mapa detallado que muestra cómo diferentes eventos o variables se relacionan y ocurren juntos. En lugar de calcular todas las posibles combinaciones de eventos de manera separada (lo cual sería muy complejo), la red bayesiana divide este cálculo en partes más pequeñas y manejables usando relaciones de causa y efecto. Por ejemplo, si queremos analizar la probabilidad de que alguien salga a correr basándonos en el clima y la temperatura, podemos dividirlo en tres partes más simples: primero calculamos la probabilidad del clima, luego la probabilidad de la temperatura dado el clima, y finalmente la probabilidad de salir a correr dadas las condiciones anteriores. Esta forma de descomponer probabilidades complejas en partes más simples hace que los cálculos sean más prácticos y fáciles de entender, permitiendo modelar situaciones del mundo real de manera más eficiente.

Teorema de Bayes: Las redes bayesianas se basan en el teorema de *Bayes* para actualizar las probabilidades de los estados de las variables a medida que se obtiene nueva evidencia. El teorema de Bayes nos permite calcular la probabilidad de un evento desconocido basándonos en información previa o condicional. Se formula como: *la probabilidad de un evento **A** dado que ocurrió otro evento **B** es igual a la probabilidad de **B** dado **A**, multiplicada por la probabilidad de **A**, y dividida por la probabilidad de **B***. Esta fórmula matemática puede expresarse como $P(A|B) = (P(B|A) \times P(A)) / P(B)$, donde el símbolo | significa "dado que ocurrió". En otras palabras, el teorema nos ayuda a actualizar nuestras creencias sobre la probabilidad de un evento cuando tenemos nueva información relacionada con ese evento. Se que suena muy difícil, pero en un momento lo veremos en un ejemplo.

Para entender esto de manera intuitiva, pensá en cómo actualizamos nuestras creencias en la vida diaria: si ves nubes oscuras (evento B), actualizas tu creencia sobre la probabilidad de que llueva (evento A). Tu intuición naturalmente considera tanto la frecuencia con la que llueve cuando hay nubes oscuras, como la frecuencia general de lluvia en tu área. El teorema de Bayes simplemente formaliza matemáticamente este proceso de pensamiento que realizamos de manera natural.

Inferencia: La inferencia en redes bayesianas es el proceso por el cual podemos hacer predicciones y obtener respuestas basadas en la información que tenemos disponible. Imaginá que tenés una red bayesiana que modela el diagnóstico de enfermedades: cuando un paciente llega con ciertos síntomas (esta es tu evidencia), la inferencia te permite calcular la probabilidad de que tenga diferentes enfermedades. El sistema toma en cuenta todas las relaciones entre síntomas y enfermedades que están modeladas en la red, y usando el teorema de Bayes, actualiza las probabilidades para darte las predicciones más precisas posibles. Por ejemplo, si un paciente llega con fiebre y dolor de garganta, la inferencia nos permitiría calcular qué tan probable es que tenga gripe versus un resfriado común, considerando también otros factores como la época del año o la edad del paciente.



Fundamentos de las Redes Bayesianas

Nodos

Representan variables aleatorias con estados y distribución de probabilidad.

Aristas

Indican dependencias condicionales entre variables.

Distribución de Probabilidad

Codifica la distribución conjunta de todas las variables.

Teorema de Bayes

Actualiza probabilidades con nueva evidencia.

Construcción de una Red Bayesiana

La construcción de una red bayesiana es un proceso sistemático y riguroso que permite modelar relaciones probabilísticas entre variables para hacer predicciones informadas. Este proceso es especialmente valioso en situaciones donde necesitamos tomar decisiones bajo incertidumbre, como en el caso que veremos a continuación con un ejemplo de agricultura, donde múltiples factores interactúan de manera compleja para determinar el rendimiento de los cultivos. A través de cuatro pasos fundamentales, podemos crear un modelo que capture estas interacciones y nos ayude a tomar mejores decisiones basadas en datos.

En general, estos suelen ser los pasos para construir una Red Bayesiana:

1. **Definir variables**: Identificar las variables relevantes para el problema a resolver y definir sus posibles estados. Por ejemplo, si estamos construyendo una red para predecir el rendimiento de un cultivo, las variables podrían ser: "tipo de suelo" (arcilloso, arenoso, limoso), "nivel de lluvia" (bajo, medio, alto), "temperatura" (fría, templada, cálida) y "rendimiento del cultivo" (bajo, medio, alto).

ID Registro	Fecha	Tipo de Suelo	Temperatura	Nivel de Lluvia	Rendimiento	Ton/Hectárea
001	2024-01-15	Limoso	Templada	Alta	Alto	8.5
002	2024-01-15	Arcilloso	Templada	Media	Alto	7.8
003	2024-01-15	Arenoso	Cálida	Baja	Bajo	3.2
004	2024-01-15	Limoso	Templada	Media	Alto	8.1
005	2024-01-16	Arcilloso	Fría	Alta	Medio	5.9
006	2024-01-16	Arenoso	Cálida	Media	Medio	4.8
007	2024-01-16	Limoso	Templada	Alta	Alto	8.7
008	2024-01-16	Arcilloso	Templada	Media	Alto	7.5
009	2024-01-17	Arenoso	Fría	Alta	Medio	5.2
010	2024-01-17	Limoso	Cálida	Baja	Medio	6.1
011	2024-01-17	Arcilloso	Templada	Media	Alto	7.9
012	2024-01-17	Arenoso	Cálida	Baja	Bajo	3.5

Muestra de 20 registros históricos de cultivos que servirían como base para calcular las probabilidades de la red bayesiana. En un caso real, se necesitarían cientos o miles de registros para obtener estimaciones confiables.

Criterios de Clasificación

Rendimiento por Tonelada/Hectárea:

- Bajo: < 4.5 ton/ha
- Medio: 4.5 - 7.0 ton/ha
- Alto: > 7.0 ton/ha

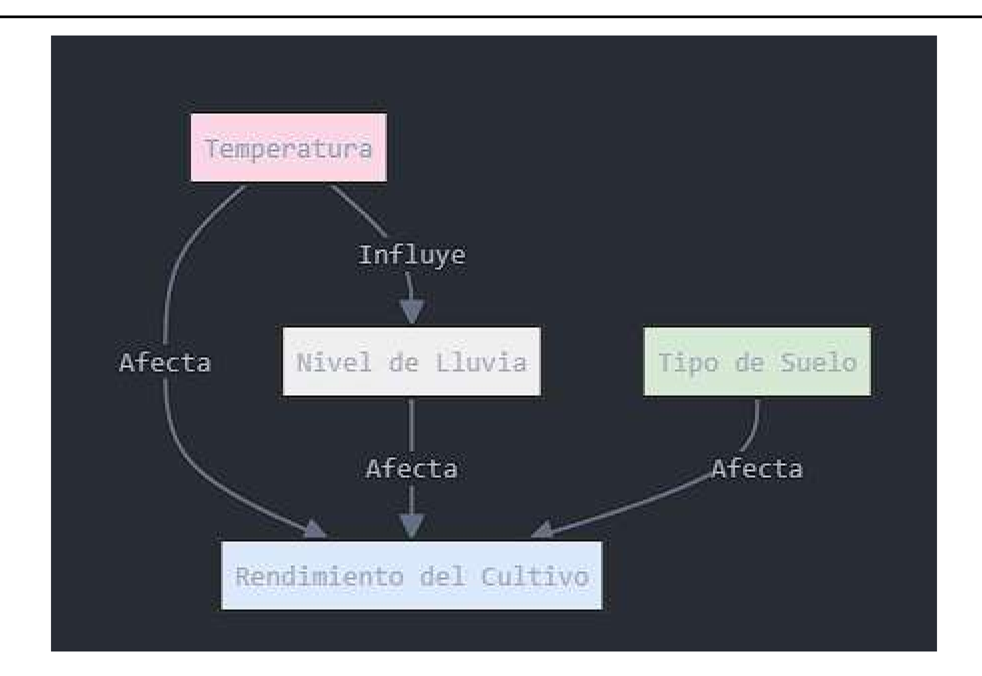
Nivel de Lluvia (mm/mes):

- Bajo: < 50 mm
- Medio: 50 - 100 mm
- Alto: > 100 mm

Cómo usar estos datos:

- ❖ Contar la frecuencia de cada valor en las variables individuales para obtener las probabilidades iniciales (por ejemplo, contar cuántas veces aparece cada tipo de suelo)
- ❖ Para las probabilidades condicionales, contar la frecuencia de combinaciones específicas (por ejemplo, cuántas veces hay lluvia alta cuando la temperatura es templada)
- ❖ Para el rendimiento, analizar las combinaciones de todas las variables y su resultado (por ejemplo, suelo limoso + temperatura templada + lluvia alta → rendimiento alto)

2. **Estructura del grafo**: Determinar las relaciones de dependencia entre las variables y construir el grafo dirigido acíclico que representa estas relaciones. Siguiendo con el ejemplo del cultivo, podríamos establecer que tanto el tipo de suelo como el nivel de lluvia y la temperatura afectan directamente al rendimiento del cultivo, y que además la temperatura influye en el nivel de lluvia.



3. **Asignación de probabilidades**: Estimar las distribuciones de probabilidad condicionales para cada variable dado sus padres en el grafo. Por ejemplo, necesitaríamos definir la probabilidad de obtener un rendimiento alto del cultivo dado un suelo arcilloso, lluvia alta y temperatura templada. Estas probabilidades pueden obtenerse de registros históricos de cosechas o consultas a agricultores expertos.

Asignación de probabilidades condicionales

Ejemplo de cómo calcular las probabilidades condicionales usando los datos históricos.

Paso 1: Extraer Datos Relevantes

Temperatura	Nivel de Lluvia	Frecuencia
Templada	Alta	3
Templada	Media	4
Templada	Baja	1

Paso 2: Calcular Probabilidades Condicionales

P(Lluvia | Temperatura = Templada)

- **Alta:** 3/8 = 0.375 (37.5%)
- **Media:** 4/8 = 0.500 (50.0%)
- **Baja:** 1/8 = 0.125 (12.5%)

Paso 3: Tabla de Probabilidad Condicional Completa

Temperatura	P(Lluvia Alta)	P(Lluvia Media)	P(Lluvia Baja)
Templada	0.375	0.500	0.125
Cálida	0.200	0.300	0.500
Fría	0.400	0.400	0.200

Ejemplo de Probabilidad de Rendimiento

Tipo de Suelo	Temperatura	Nivel de Lluvia	P(Rendimiento Alto)	Cálculo
Limoso	Templada	Alta	0.85	3 de 3 casos similares (85%)

Notas Importantes:

- Las probabilidades se calculan dividiendo la frecuencia de cada combinación entre el total de casos para esa condición.
- Para probabilidades condicionales con múltiples variables (como el rendimiento), se consideran todas las combinaciones relevantes.
- En casos reales, se necesitarían más datos para obtener estimaciones más confiables.
- Las probabilidades deben sumar 1 (100%) para cada conjunto de condiciones.

4. **Validación del modelo**: Validar la red bayesiana utilizando datos adicionales para asegurar que el modelo represente adecuadamente las relaciones y las dependencias en el sistema. Por ejemplo, podríamos usar datos de cosechas de años anteriores que no se utilizaron en la construcción del modelo para verificar si las predicciones de la red coinciden con los resultados reales observados.

ID	Tipo de Suelo	Temperatura	Nivel de Lluvia	Rendimiento Real	Predicción Modelo	¿Coincide?
T001	Limoso	Templada	Alta	Alto	Alto (85%)	✓
T002	Arcilloso	Templada	Media	Alto	Alto (70%)	✓
T003	Arenoso	Cálida	Baja	Medio	Bajo (65%)	✗
T004	Limoso	Fría	Alta	Medio	Medio (55%)	✓
T005	Arcilloso	Cálida	Baja	Bajo	Bajo (75%)	✓

Métricas de Validación

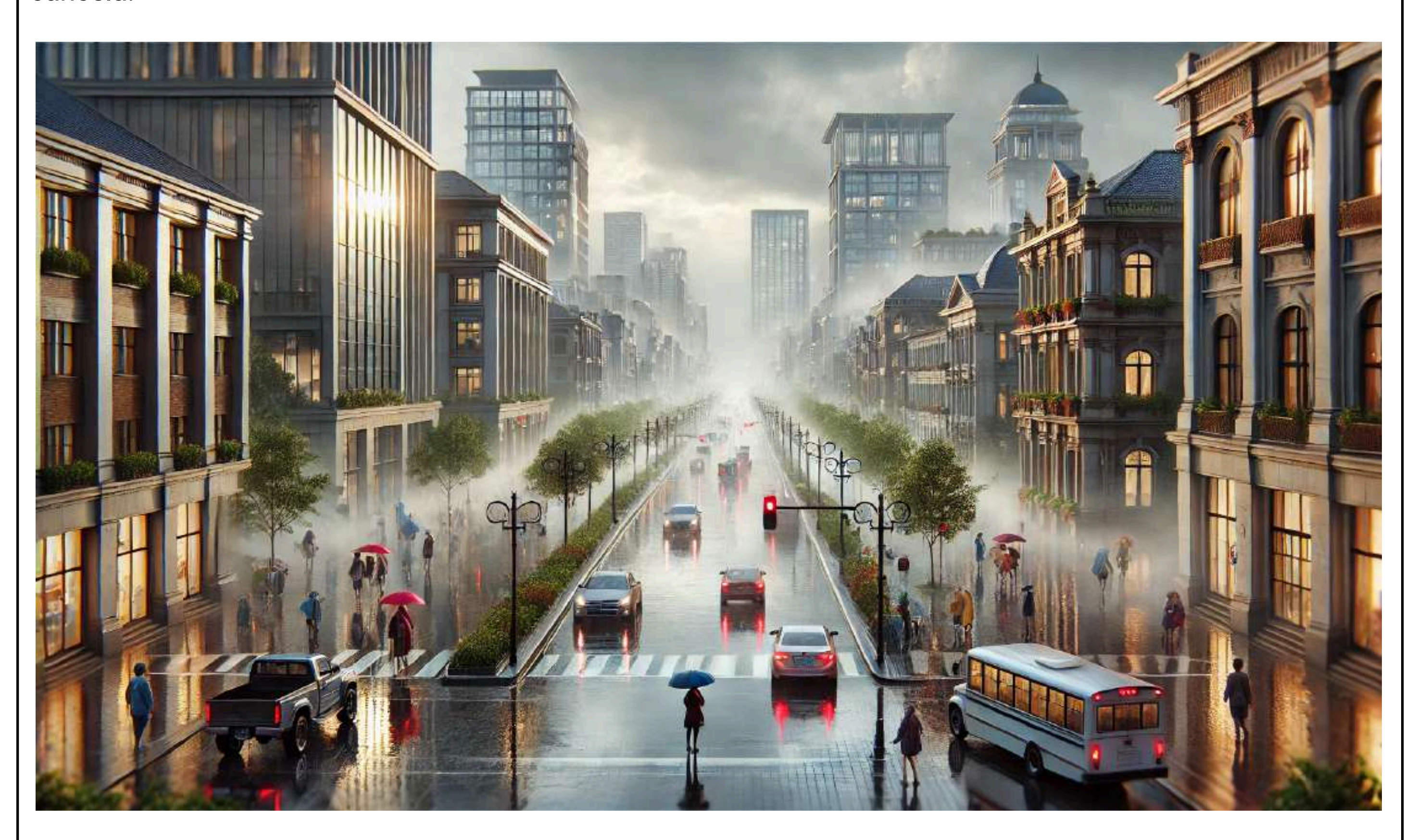
- Precisión del Modelo
  - Predicciones correctas: 4
  - Total de predicciones: 5
  - Precisión: 80%
- Distribución de Errores
  - Sobreestimaciones: 0
  - Subestimaciones: 1
  - Error más común: Predicción Bajo cuando fue Medio

Como podés imaginarte, la construcción de una red bayesiana es un proceso metodológico que necesita un balance entre el conocimiento del dominio y el análisis de datos. Comenzando con la identificación cuidadosa de las variables relevantes, siguiendo con el establecimiento de relaciones causales en la estructura del grafo, continuando con la asignación precisa de probabilidades condicionales, y finalizando con una validación rigurosa, cada paso construye sobre el anterior para crear un modelo robusto y útil. Este enfoque estructurado permite desarrollar modelos que no solo capturan las dependencias probabilísticas entre variables, sino que también proporcionan un marco interpretable para la toma de decisiones bajo incertidumbre. La clave del éxito en la construcción de redes bayesianas está en mantener un equilibrio entre la complejidad del modelo y su capacidad para representar fielmente el sistema que se está

modelando, siempre teniendo en cuenta que el modelo final debe ser tanto preciso en sus predicciones como útil en su aplicación práctica.

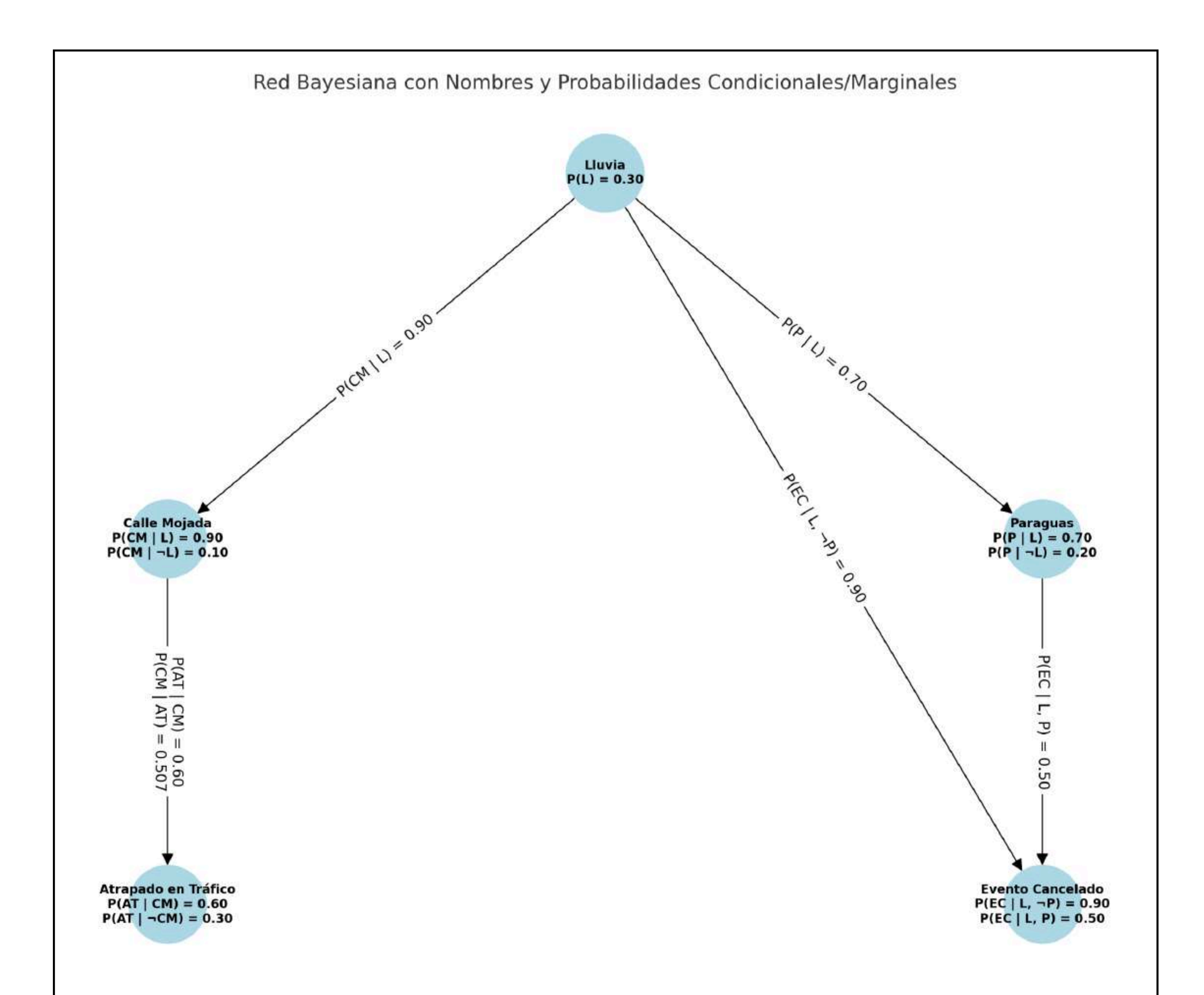
Ejemplo de inferencia con una red bayesiana

A continuación, vamos a analizar un ejemplo donde la lluvia afecta varios aspectos de la vida diaria: si las calles están mojadas, si alguien usa paraguas, si el tráfico se ve afectado y si un evento al aire libre se cancela.



Nodos y dependencias:

1. **Lluvia** 🌧️: Es el nodo causal principal de nuestro ejemplo. Si llueve, tiene implicaciones para varios eventos relacionados.
2. **Calle Mojada** 💧: Este nodo está directamente influenciado por la lluvia. Si llueve, es más probable que las calles estén mojadas.
3. **Paraguas** ☂️: Este nodo también depende de si está lloviendo. Si llueve, las personas probablemente lleven paraguas para protegerse.
4. **Atrapado en Tráfico** 🚗: El hecho de que alguien se quede atrapado en tráfico está condicionado a que las calles estén mojadas, lo que puede dificultar el flujo de vehículos y aumentar la congestión.
5. **Evento Cancelado** ❌: Finalmente, el nodo que representa la cancelación de un evento depende tanto de la lluvia como del uso de paraguas. Si está lloviendo, y especialmente si las personas no llevan paraguas, es más probable que un evento al aire libre sea cancelado.



Inferencia Bayesiana: probabilidad de *calle mojada* dado *atrapado en tráfico*

En este ejemplo, queremos aplicar *inferencia* y calcular la probabilidad de que las **calles estén mojadas** sabiendo que alguien está **atrapado en tráfico**.

Para reflejar la resolución de este ejemplo en el gráfico, vamos a visualizar cómo aplicamos la inferencia bayesiana para calcular la **probabilidad de *Calle Mojada* dado *Atrapado en Tráfico***.

Antes de hacer cualquier cálculo, nuestra intuición nos diría que si alguien está atrapado en el tráfico, hay una alta probabilidad de que las calles estén mojadas, ya que sabemos que el mal estado de las calles suele causar congestión vehicular. Esta intuición básica se alinea con el razonamiento probabilístico que realizamos inconscientemente en nuestra vida diaria, donde asociamos naturalmente el tráfico intenso con condiciones climáticas adversas. El cálculo bayesiano nos ayudará a cuantificar exactamente esta intuición.

La probabilidad de que las calles estén mojadas dado que estás atrapado en tráfico **P(CM/AT)** se calcula con el **Teorema de Bayes**:

$$P(\text{CM}|\text{AT}) = \frac{P(\text{AT}|\text{CM}) \cdot P(\text{CM})}{P(\text{AT})}$$

Esta fórmula se lee como:

"La probabilidad de Calle Mojada (CM) dado Atrapado en Tráfico (AT) es igual a la probabilidad de Atrapado en Tráfico dado que hay Calle Mojada, multiplicado por la probabilidad de Calle Mojada, todo esto dividido por la probabilidad de Atrapado en Tráfico"

O de manera más coloquial:

"La probabilidad de que la calle esté mojada, sabiendo que hay tráfico, es igual a la probabilidad de que haya tráfico cuando la calle está mojada, multiplicada por la probabilidad general de que la calle esté mojada, dividido todo por la probabilidad general de que haya tráfico"

Datos (los encontramos en nuestra red hecha en la página anterior):

- $P(\text{AT}|\text{CM}) = 0.60$: Probabilidad de estar atrapado en tráfico si las calles están mojadas.
- $P(\text{CM}|L) = 0.90$, $P(\text{CM}|\neg L) = 0.10$: Probabilidad de calles mojadas si llueve o no.
- $P(L) = 0.30$: Probabilidad de lluvia.

Cálculos que necesitamos hacer con los datos de arriba:

1. **Cálculo de $P(\text{CM})$:**
   Esta es la probabilidad total de que las calles estén mojadas, considerando tanto si llueve como si no:

$$P(\text{CM}) = 0.90 \cdot 0.30 + 0.10 \cdot 0.70 = 0.34$$

Es decir, que la probabilidad de que las calles estén mojadas depende de la probabilidad de que estén mojadas (90%) cuando llueve (30%) y que estén mojadas (10%) cuando no llueve (70%). Eso resulta en un 34% de probabilidad. Ahora nos queda por calcular P(AT).

2. **Cálculo de $P(\text{AT})$:**
   La probabilidad total de estar atrapado en tráfico, considerando calles mojadas y no mojadas:

$$P(\text{AT}) = 0.60 \cdot 0.34 + 0.30 \cdot 0.66 = 0.402$$

Es decir, que la probabilidad de estar atrapado en el tráfico depende de la probabilidad de estar atrapado en el tráfico (60%) cuando las calles estén mojadas (34%) y de estar atrapado en el tráfico (30%) cuando no estén mojadas (66%). Esto resulta en un 40.2% de probabilidad. Con esto ya tenemos todo lo que necesitamos para aplicar el teorema de Bayes y calcular la probabilidad de que las calles estén mojadas

dado que sabemos que estamos atrapados en el tráfico.

Finalmente, aplicamos Bayes:

$$P(\mathrm{CM}|\mathrm{AT}) = \frac{0.60 \cdot 0.34}{0.402} \approx 0.507$$

Utilizando estos valores, calculamos que la probabilidad de que las **calles estén mojadas** dado que sabemos que estamos atrapados en el **tráfico** es del **50.7%**. Esto significa que, si observamos congestión vehicular, hay una posibilidad considerable de que las calles estén mojadas, lo que puede ser un factor en el tráfico.



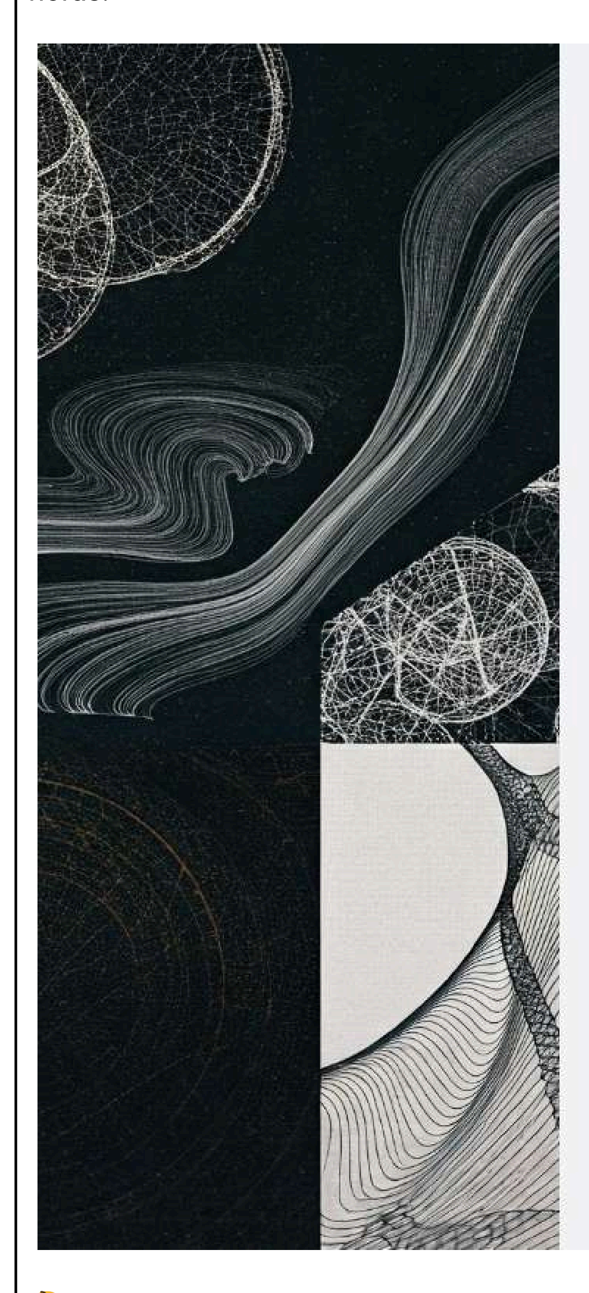
Aplicaciones de las Redes Bayesianas

🩺 **Diagnóstico Médico:** Las redes bayesianas se utilizan para modelar la relación entre síntomas y enfermedades, ayudando a los médicos a diagnosticar condiciones médicas y predecir la probabilidad de enfermedades basadas en la evidencia clínica. Por ejemplo, si un paciente presenta fiebre (90% de los casos de gripe), dolor muscular (80% de los casos de gripe) y congestión nasal (70% de los casos de gripe), la red puede calcular que la probabilidad de que tenga gripe es alta (85%), mientras que la probabilidad de un resfriado común sería menor (35%).

👍 **Sistemas de Recomendación:** En los sistemas de recomendación, las redes bayesianas pueden modelar las preferencias de los usuarios y las dependencias entre diferentes ítems, proporcionando recomendaciones personalizadas. Por ejemplo, si un usuario vió películas de ciencia ficción con altas calificaciones (80% de probabilidad de gusto por el género) y específicamente películas de Christopher Nolan (90% de probabilidad de apreciación), la red podría recomendar "Interstellar" con una alta confianza

(85% de probabilidad de que le guste).

🔧 **Predicción de Fallos:** En la industria, las redes bayesianas se emplean para predecir fallos en sistemas técnicos y equipos, permitiendo el mantenimiento predictivo y la gestión de riesgos. Por ejemplo, si un motor muestra vibraciones anormales (presente en 75% de los fallos) y temperatura elevada (presente en 85% de los fallos), la red puede calcular una probabilidad del 90% de fallo inminente en las próximas 24 horas.



Aplicaciones de las Redes Bayesianas



Diagnóstico Médico

Modelan relación entre síntomas y enfermedades, como el diagnóstico de enfermedades cardíacas a partir de síntomas y resultados de exámenes.



Predicción de Fallos

Predicen fallos en sistemas técnicos y equipos, como el mantenimiento predictivo de turbinas eólicas.



Bioinformática

Modelan interacciones genéticas y analizan datos genómicos, como la identificación de riesgos de enfermedades hereditarias.

🤔 **Toma de Decisiones:** Las redes bayesianas son herramientas valiosas para la toma de decisiones en situaciones de incertidumbre, como la gestión de proyectos, la planificación estratégica y la evaluación de riesgos. Por ejemplo, en la decisión de lanzar un producto, la red puede considerar factores como el interés del mercado (70% favorable), la competencia actual (30% de mercado saturado) y los costos de producción (60% dentro del presupuesto) para calcular una probabilidad del 65% de éxito comercial.

🧬 **Bioinformática:** En bioinformática, las redes bayesianas se utilizan para modelar interacciones genéticas, predecir la función de genes y analizar datos genómicos. Por ejemplo, si se observa la expresión elevada de ciertos genes reguladores (presentes en el 80% de los casos de resistencia a medicamentos) y patrones

específicos de mutación (presentes en el 70% de los casos), la red puede predecir una probabilidad del 75% de que un tumor desarrollará resistencia a un tratamiento específico.

🔍 **Reconocimiento de Patrones:** Las redes bayesianas se aplican en el reconocimiento de patrones y la clasificación, donde modelan las relaciones entre características observables y categorías latentes. Por ejemplo, en el reconocimiento de spam, si un email contiene palabras clave sospechosas (presente en 90% del spam), enlaces dudosos (presente en 85% del spam) y está enviado en horario inusual (presente en 70% del spam), la red puede calcular una probabilidad del 95% de que sea spam.

Ventajas de las Redes Bayesianas

Manejo de la Incertidumbre: Las redes bayesianas proporcionan un marco coherente para manejar la incertidumbre y la variabilidad en sistemas complejos, permitiendo la toma de decisiones informadas. Por ejemplo, en un diagnóstico médico, pueden manejar situaciones donde los síntomas son ambiguos o incompletos, asignando probabilidades a diferentes diagnósticos posibles en lugar de dar respuestas definitivas. Esta capacidad es especialmente valiosa en situaciones del mundo real donde rara vez tenemos certeza absoluta.

Modularidad: La estructura modular de las redes bayesianas facilita la actualización y la expansión del modelo a medida que se dispone de nueva información o cambia el dominio del problema. Por ejemplo, en un sistema de diagnóstico automotriz, se pueden añadir fácilmente nuevos componentes o relaciones sin necesidad de reconstruir todo el modelo. Si surge un nuevo tipo de sensor o una nueva relación entre componentes, simplemente se puede agregar como un nuevo nodo o conexión en la red existente.

Inferencia Probabilística: Permiten realizar inferencias probabilísticas eficientes, calculando la probabilidad de variables no observadas dado un conjunto de evidencias observadas. Por ejemplo, en un sistema de predicción meteorológica, pueden inferir la probabilidad de lluvia basándose en múltiples factores como presión atmosférica, humedad y temperatura, incluso cuando algunos de estos datos faltan. La red puede "razonar" en múltiples direcciones, desde causas a efectos o viceversa.

Transparencia y Explicabilidad: Las redes bayesianas son transparentes y explicables, proporcionando una representación visual clara de las relaciones entre variables y permitiendo interpretar fácilmente los resultados. A diferencia de las "cajas negras" como las redes neuronales profundas, las redes bayesianas muestran explícitamente cómo se relacionan las variables y cómo se llega a las conclusiones. Un experto puede examinar la red, entender su estructura y verificar si las relaciones modeladas tienen sentido desde el punto de vista del dominio. Por ejemplo, en un sistema de evaluación de riesgos financieros, los analistas pueden ver y entender exactamente qué factores influyen en las predicciones y cómo lo hacen.

Con esto concluimos nuestra exploración de los sistemas expertos basados en conocimiento y razonamiento probabilístico. Hasta ahora, hemos cubierto desde los fundamentos básicos de la IA y el *machine learning* hasta metodologías más estructuradas como los sistemas expertos y las redes bayesianas. Nos quedan solo tres *conceptos básicos* más pero implica meternos en las fronteras más avanzadas y emocionantes de la inteligencia artificial.

Los *transformers* y *LLMs* representan una revolución en el procesamiento del lenguaje natural, llevando la comprensión y generación de texto a niveles sin precedentes.

La *IA basada en agentes* nos introduce en el hermoso mundo de los sistemas autónomos que pueden percibir, razonar y actuar de forma independiente.

Finalmente, la *robótica* nos mostrará cómo estos conceptos de la IA se materializan en el mundo físico, combinando inteligencia artificial con ingeniería mecánica.

11. 🚀 Transformers y LLMs

Transformers y Modelos de Lenguaje de Gran Escala

Los transformers y LLMs han revolucionado el procesamiento del lenguaje natural. Han mejorado la comprensión y generación de texto por máquinas, permitiendo nuevas aplicaciones en NLP.



Al estar leyendo este libro, es probable que tu interés por la inteligencia artificial haya surgido en alguna medida gracias al *boom* en el campo que se generó con el nacimiento de los LLM, puntualmente desde la aparición de ChatGPT, que puso al alcance de todas las personas por primera vez una aplicación con la cual interactuar con una inteligencia artificial nunca antes vista.

Este suceso disparó la imaginación de lo que es posible y generó una revolución espectacular donde todas las disciplinas se encuentran prácticamente obligadas a revisar cómo la interacción con estos modelos de IA les impactará en mayor o menor medida, para bien o para mal (yo opino que para bien en la gran mayoría de casos).

Recuerdo muy bien por Febrero del año 2023 estar liderando un *datathon* en una escuela de negocios en la ciudad de Valencia en el cual se desafiaba a los alumnos y alumnas a conceptualizar e idear nuevos negocios a través de la aplicación de inteligencia artificial. Lo que no me esperaba para esa fecha era que ya hubiera equipos utilizando activamente ChatGPT para literalmente resolver el desafío en un par de horas y ganarlo. Su uso no estaba prohibido ni mucho menos, tampoco había razón para hacerlo, el 70% de los alumnos aún no habían tocado ChatGPT. El punto es que un desafío que en general tomaría al menos un día entero de trabajo de un equipo de no menos de cinco personas era resuelto en poco más de dos horas por un par de ellos únicamente. No me quedó otra opción que aceptar que los

desafíos deberían subir de nivel y que el volúmen de trabajo y de resultados demandados debía adaptarse de ahora en adelante, incluyendo la forma de evaluar a los equipos. Quedé verdaderamente impresionado, pero para bien.

En economía esto se conoce como una **revolución tecnológica**. Revoluciones como la que estamos viviendo han habido muchas veces en la historia y permiten producir más bienes y servicios con la misma cantidad de recursos o incluso con menos de ellos, lo que aumenta la productividad laboral y la productividad del capital.

Puntualmente, en el corazón de esta revolución actual, se encuentra una pieza de arquitectura de software conocida como **Transformer**. Tanto en BERT como en GPT como en Claude, Gemini, Llama (entre otros) existe un Transformer como piedra angular de una gran red neuronal diseñada para aprender nuestro lenguaje y replicar casi de forma idéntica cómo los humanos lo utilizamos. Nosotros ya hemos hablado de Transformers en reiteradas ocasiones en este libro, pero ahora será el protagonista de esta sección.

Los transformers y los modelos de lenguaje de gran escala (LLMs, por sus siglas en inglés) representan uno de los avances más significativos en el campo del procesamiento de lenguaje natural (NLP) y la inteligencia artificial en los últimos años. Los transformers revolucionaron la forma en que las máquinas comprenden y generan texto, permitiendo una amplia gama de aplicaciones que van desde la traducción automática y los chatbots hasta la generación de texto y la creación de contenido. Modelos como BERT, GPT, Claude, Llama, Gemini y sus sucesores han establecido nuevos estándares en la calidad y capacidad de las aplicaciones de NLP.

Los transformers son una arquitectura de red neuronal que se conocieron por primera vez con el paper "Attention is All You Need" de Vaswani et al. en 2017[81]. A diferencia de las redes neuronales recurrentes (RNN) y las redes neuronales convolucionales (CNN), los transformers utilizan mecanismos de *atención* para procesar secuencias de datos. De la misma forma que lo vimos con los Word Embeddings, los Transformers tienen una estrategia de aprender el significado semántico de cada palabra, pero además incorporan una novedad llamada *mecanismo de atención* para mejorar la forma en que el modelo entiende qué palabras se relacionan con qué otras dentro de una oración e incluso de un texto largo.

La arquitectura del transformer y el mecanismo de *atención* revolucionaron el mundo y la IA dejó de ser algo curioso para convertirse en algo *mainstream*.

A continuación la primera página del mítico paper:

[81] El paper "Attention is All You Need" fue publicado por Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser y Illia Polosukhin, investigadores de Google Brain y Google Research.

Attention Is All You Need

Ashish Vaswani* Google Brain avaswani@google.com

Noam Shazeer* Google Brain noam@google.com

Niki Parmar* Google Research nikip@google.com

Jakob Uszkoreit* Google Research usz@google.com

Llion Jones* Google Research llion@google.com

Aidan N. Gomez* † University of Toronto aidan@cs.toronto.edu

Łukasz Kaiser* Google Brain lukaszkaiser@google.com

Illia Polosukhin* ‡ illia.polosukhin@gmail.com

Abstract

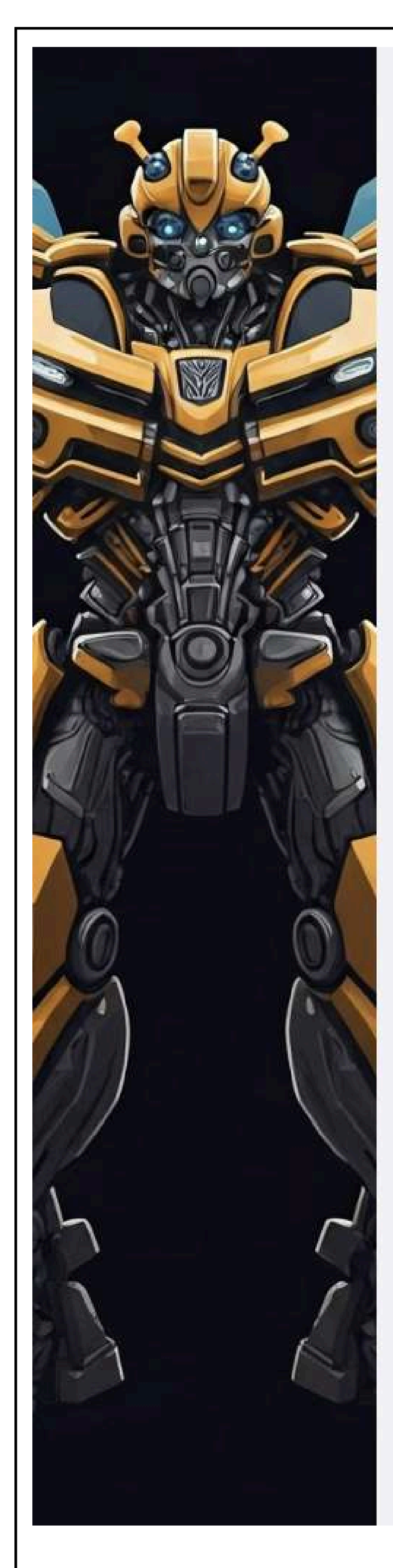
The dominant sequence transduction models are based on complex recurrent or convolutional neural networks that include an encoder and a decoder. The best performing models also connect the encoder and decoder through an attention mechanism. We propose a new simple network architecture, the Transformer, based solely on attention mechanisms, dispensing with recurrence and convolutions entirely. Experiments on two machine translation tasks show these models to be superior in quality while being more parallelizable and requiring significantly less time to train. Our model achieves 28.4 BLEU on the WMT 2014 English-to-German translation task, improving over the existing best results, including ensembles, by over 2 BLEU. On the WMT 2014 English-to-French translation task, our model establishes a new single-model state-of-the-art BLEU score of 41.8 after training for 3.5 days on eight GPUs, a small fraction of the training costs of the best models from the literature. We show that the Transformer generalizes well to other tasks by applying it successfully to English constituency parsing both with large and limited training data.

*Equal contribution. Listing order is random. Jakob proposed replacing RNNs with self-attention and started the effort to evaluate this idea. Ashish, with Illia, designed and implemented the first Transformer models and has been crucially involved in every aspect of this work. Noam proposed scaled dot-product attention, multi-head attention and the parameter-free position representation and became the other person involved in nearly every detail. Niki designed, implemented, tuned and evaluated countless model variants in our original codebase and tensor2tensor. Llion also experimented with novel model variants, was responsible for our initial codebase, and efficient inference and visualizations. Lukasz and Aidan spent countless long days designing various parts of and implementing tensor2tensor, replacing our earlier codebase, greatly improving results and massively accelerating our research.

†Work performed while at Google Brain.

‡Work performed while at Google Research.

31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.



Fundamentos de los Transformers

Principio de los Transformers

Los **transformers** son un tipo de modelo de aprendizaje profundo que se ha convertido en uno de los enfoques más importantes para el procesamiento del lenguaje natural (NLP). La clave de su éxito radica en la **atención**, un mecanismo que permite al modelo enfocar su atención en las partes más relevantes de la secuencia de entrada, en lugar de procesar toda la información de manera uniforme.

La Auto-Atención

La **auto-atención** es un componente fundamental de los **transformers**. Este mecanismo compara cada palabra de la entrada con las demás, permitiendo al modelo capturar dependencias a largo plazo entre las palabras. Esto mejora significativamente la capacidad del modelo para comprender el contexto y significado del lenguaje.

Arquitectura de los Transformers

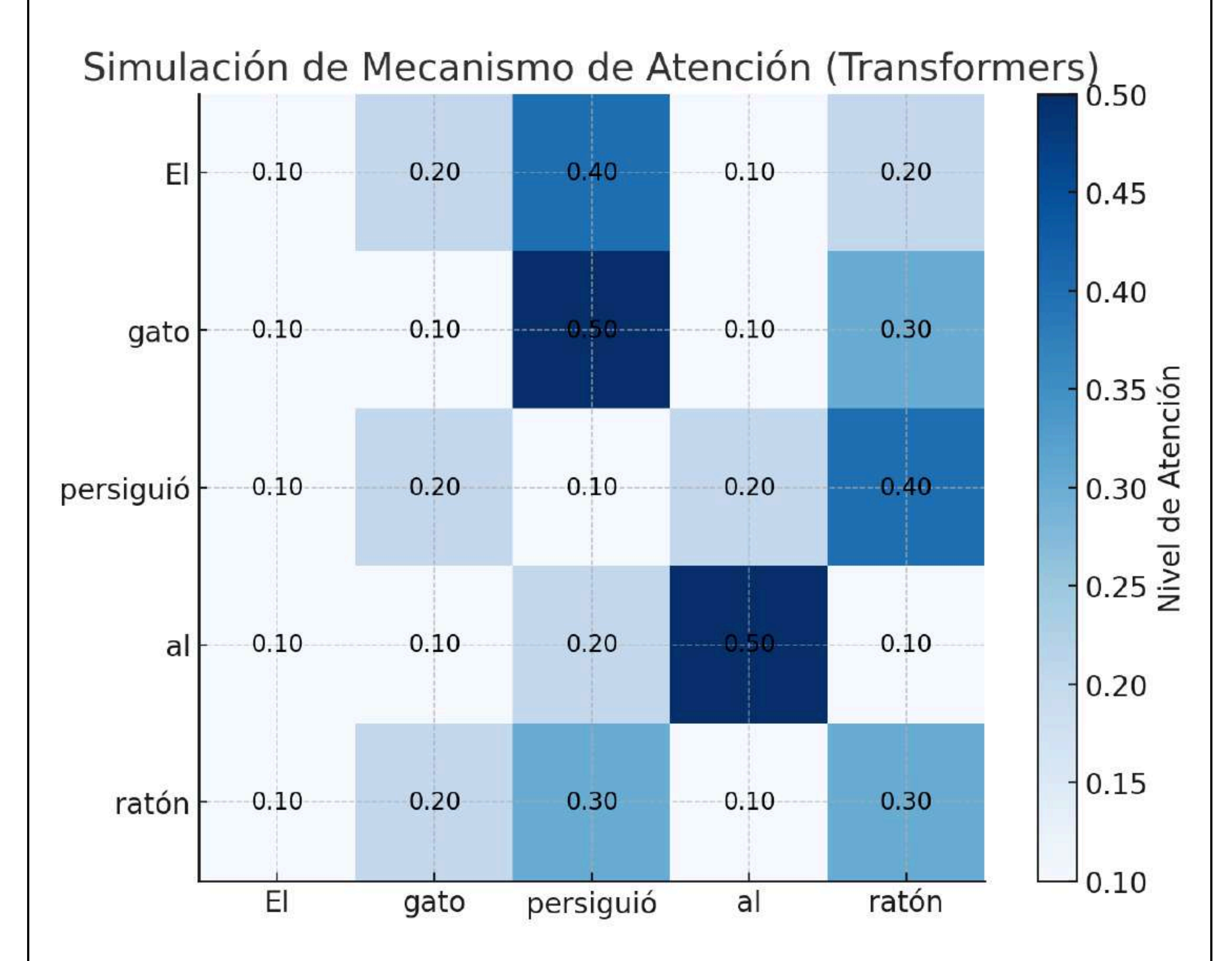
Las capas de **transformador** están compuestas por codificadores y decodificadores que utilizan múltiples **cabezas de atención**. Estas capas profundas permiten a los **transformers** aprender representaciones lingüísticas complejas y abstractas, lo que los convierte en herramientas poderosas para tareas como traducción, resumen y generación de texto.

Fundamentos de los Transformers

Atención: El mecanismo de atención en los transformers funciona como un sistema que permite al modelo "prestar atención" a diferentes partes del texto de manera selectiva, similar a cómo los humanos nos enfocamos en diferentes partes de una frase para entender su significado. Por ejemplo, en la frase "Juan, quien vive en Madrid desde hace años, ama la comida española", cuando queremos entender sobre quién estamos hablando, prestamos más atención a "Juan", pero cuando queremos saber qué le gusta, nos enfocamos en "ama la comida española".

Para lograr esto, para cada palabra o token que está procesando, el modelo calcula tres vectores: consulta (lo que estamos buscando), clave (dónde buscarlo) y valor (qué información extraer). Es como si cada palabra tuviera una "llave" (clave) que puede abrir diferentes "cerraduras" (consultas) con distinta facilidad, determinando así cuánta atención se debe prestar a cada parte del texto.

El siguiente gráfico muestra cómo cada palabra en la oración "El gato persiguió al ratón" pone atención en las demás. Por ejemplo, "gato" pone más atención en "persiguió" y "ratón", que son palabras claves en el contexto de la acción.



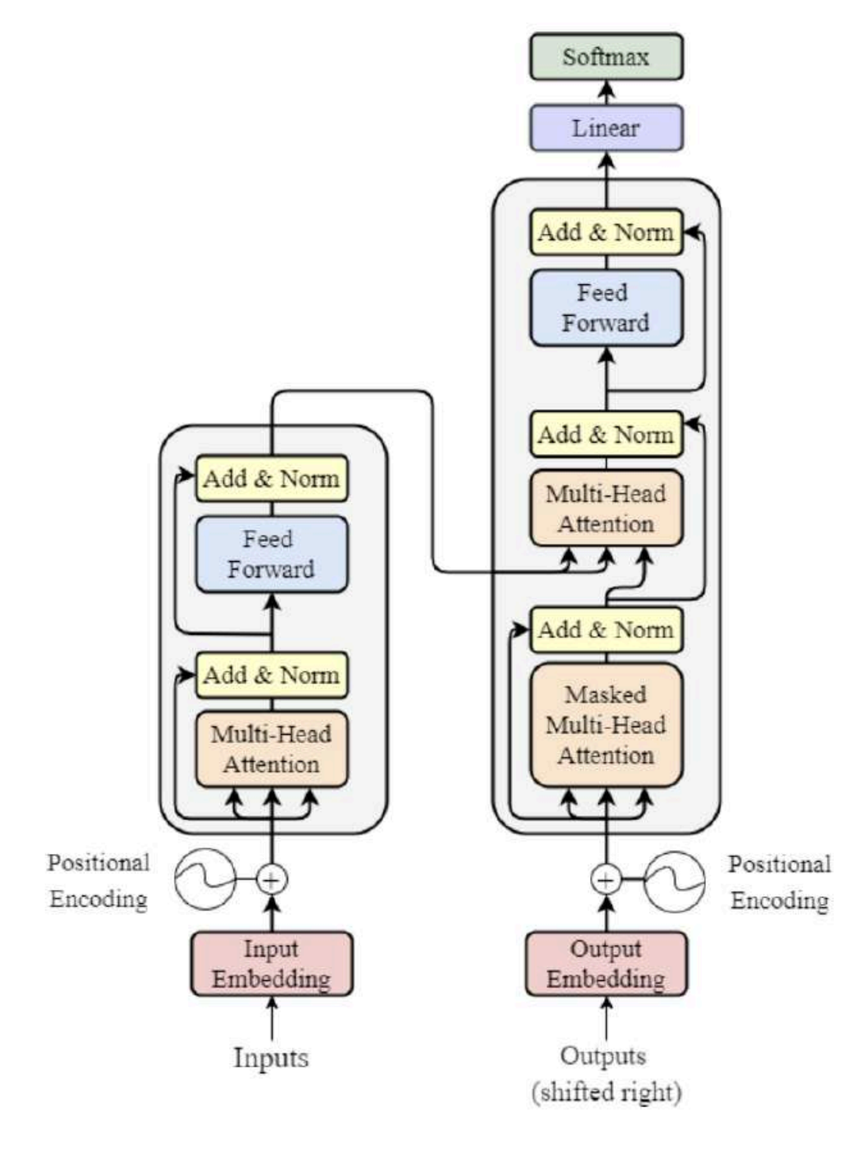
Los Transformers utilizan estos mecanismos de atención para entender cómo cada palabra en una oración se relaciona con otras, sin importar su posición. Esto permite que el modelo capte mejor el contexto y el significado de las palabras. Gracias a esto, los Transformers son muy útiles en tareas como la traducción automática, el análisis de texto y la generación de respuestas, ya que pueden enfocarse en las palabras más importantes y entender mejor el lenguaje.

Auto-Atención (Self-Attention): La auto-atención es una implementación específica y fundamental del mecanismo de atención, donde cada palabra en una secuencia actúa simultáneamente como consulta, clave y valor para todas las demás palabras. En otras palabras, mientras que la atención general nos permite enfocarnos en diferentes partes del texto, la auto-atención asegura que cada palabra "preste atención"

específicamente a todas las demás palabras para construir su significado contextual.

Por ejemplo, en la frase "El banco está cerrado" - para entender si nos referimos a un banco financiero o a un asiento, cada palabra necesita considerar todas las demás palabras de la oración. De manera similar, en "María le dio un regalo a Juan porque era su cumpleaños", cuando el modelo procesa la palabra "su", necesita relacionarla tanto con "María" como con "Juan" para determinar de quién es el cumpleaños. El modelo logra esto calculando un "puntaje de atención" entre cada par de palabras, permitiéndole crear una representación rica en contexto que captura incluso relaciones entre palabras que están muy separadas en la oración.

Este sistema permite al modelo establecer conexiones dinámicas entre diferentes elementos de la secuencia y capturar relaciones de largo alcance, superando así las limitaciones de los modelos secuenciales tradicionales que solo podían procesar la información de manera lineal, palabra por palabra.



A la izquierda podés ver la arquitectura completa de un Transformer tal como se lo presentó en el paper "Attention is all you need" y está dividida en dos columnas: el encoder (izquierda) y decoder (derecha).

Las entradas pasan primero por un embedding y codificación posicional, luego atraviesan bloques repetitivos que contienen mecanismos de atención (Multi-Head Attention) y redes feed-forward, con capas de normalización (Add & Norm) entre medio. El decoder agrega una capa extra de atención "enmascarada" y finaliza con una capa lineal y softmax para generar las predicciones. Las flechas laterales indican conexiones residuales que ayudan en el entrenamiento.

Para entenderlo de manera intuitiva, podemos pensar en el Transformer como un traductor experto que trabaja en dos fases: primero entiende (encoder) y luego genera (decoder). En la fase de entendimiento, el modelo analiza cada palabra en relación con todas las demás, similar a como una persona lee una oración completa para entender el contexto de cada palabra. En la fase de generación, es como si el traductor fuera construyendo la traducción palabra por palabra, pero siempre teniendo en cuenta tanto el significado completo que entendió como las palabras que ya fue generando. Las conexiones residuales actúan como "notas al margen" que ayudan a mantener presente la información importante a lo largo de todo el proceso.

Podríamos pensar en el encoder como un detective lingüístico que examina minuciosamente cada palabra y su relación con las demás. Imaginá que tenés varias lupas diferentes (las "cabezas" en Multi-Head Attention)

y cada una te permite ver las relaciones entre palabras desde un ángulo distinto: una lupa podría enfocarse en relaciones gramaticales, otra en contexto semántico, y otra en referencias temporales. Así, cuando el encoder procesa una oración, está mirándola simultáneamente desde múltiples perspectivas para construir una comprensión profunda.

El decoder, por su parte, es como un escritor meticuloso que tiene acceso a todas las notas detalladas que tomó el detective (encoder). Mientras escribe cada nueva palabra, consulta constantemente estas notas para asegurarse de que está manteniendo la coherencia y el significado original. La atención "enmascarada" en el decoder sería como cubrir las palabras que aún no escribió para asegurarse de que solo usa la información disponible hasta ese momento, similar a cómo una persona construye una oración palabra por palabra.

Las capas de normalización (Add & Norm) funcionan como puntos de control que aseguran que la información no se distorsione a medida que avanza por el modelo, similar a cómo un editor revisa y ajusta un texto para mantener su claridad y coherencia. Las conexiones residuales, por su parte, son como hilos conductores que permiten que información importante del inicio del proceso llegue directamente hasta el final, evitando que se pierda en el camino.

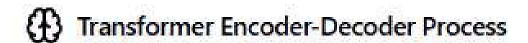
Por ejemplo, si le damos al Transformer la frase "The bank is by the river", el encoder primero analiza cada palabra en relación con las demás para entender que "bank" en este contexto se refiere a la orilla del río y no a una institución financiera. Luego, el decoder podría generar la traducción al español "La orilla está junto al río", palabra por palabra, manteniendo la coherencia del significado completo que captó el encoder. Todo esto ocurre a través de los mecanismos de atención que permiten al modelo "enfocarse" en las partes relevantes de la oración para cada decisión que toma.



Es como si tuvieras un equipo completo de lingüistas trabajando en paralelo: algunos analizando la

gramática, otros el contexto, otros las relaciones entre palabras, y todos compartiendo sus descubrimientos para construir una comprensión completa antes de comenzar a generar la traducción. Las diferentes capas y componentes del Transformer permiten que toda esta información se procese y combine de manera eficiente, produciendo resultados que pueden capturar incluso los matices más sutiles del lenguaje.

Veamos en la práctica cómo un Transformer traduce una oración del inglés al español. Tomemos como ejemplo "The cat chases the mouse" que se traducirá a "El gato persigue al ratón". El proceso comienza con la tokenización, donde la oración se divide en unidades básicas o tokens: "The", "cat", "chases", "the" y "mouse". El encoder transforma cada token en vectores numéricos que representan su significado (los embeddings que ya vimos). Por ejemplo, "cat" se convierte en [0.8, 0.3] (en la realidad ya sabés que no hay únicamente 2 dimensiones sino N, 300 por ejemplo). Luego, utilizando self-attention, el encoder crea representaciones contextuales donde cada palabra obtiene un nuevo vector que considera su relación con las demás palabras de la oración. Así, el vector de "cat" se actualiza a [0.7, 0.4], incorporando información sobre su rol en la frase.



1. Input Tokenization

The cat chases the mouse

2. Encoder Processing

Cada token se convierte en vectores que representan su significado

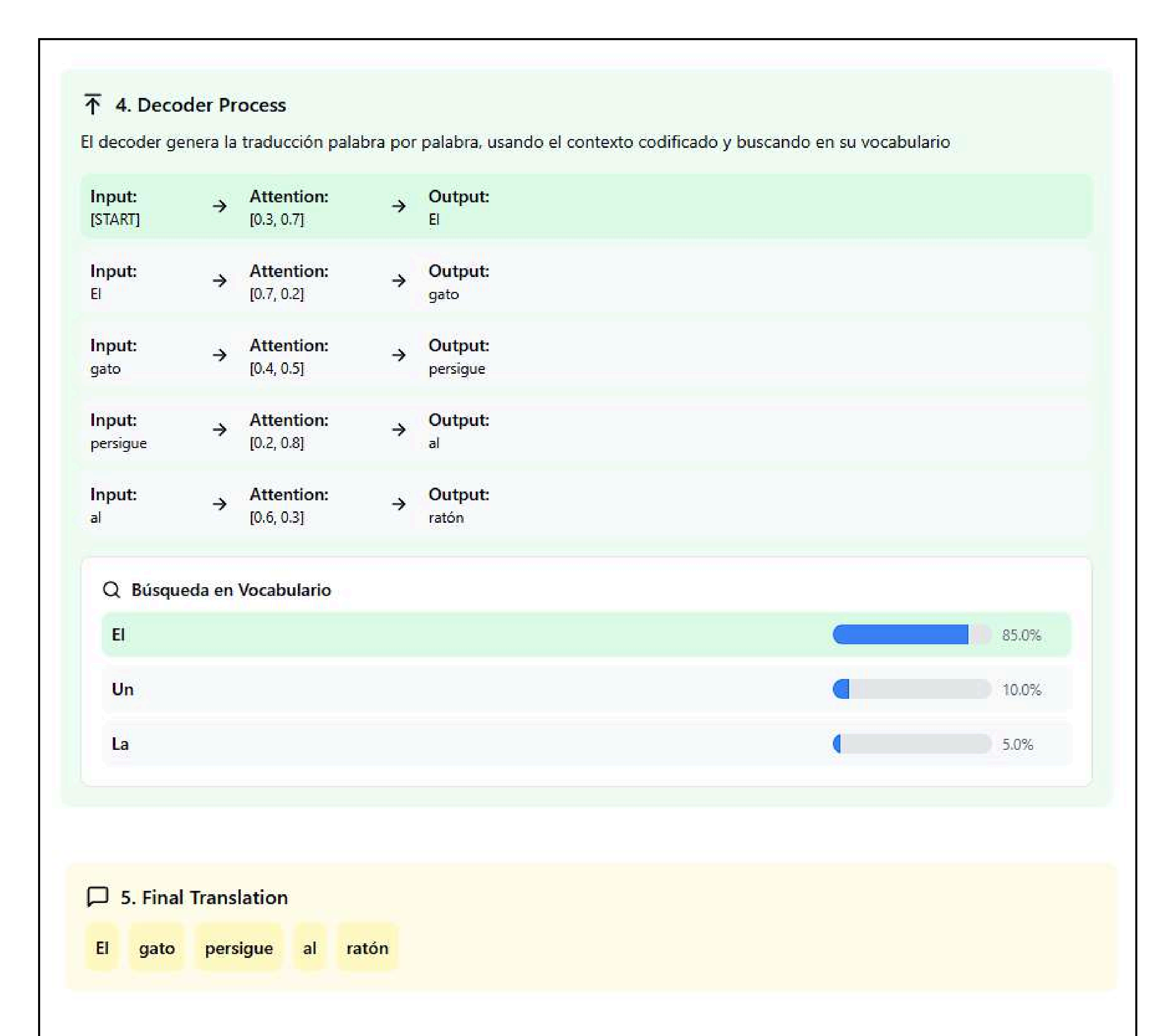
The	cat	chases	the	mouse
[0.2, 0.1]	[0.8, 0.3]	[0.5, 0.6]	[0.2, 0.1]	[0.7, 0.4]

3. Context Generation

El encoder crea representaciones contextuales usando self-attention

The	cat	chases	the	mouse
[0.2, 0.1]	[0.8, 0.3]	[0.5, 0.6]	[0.2, 0.1]	[0.7, 0.4]
[0.3, 0.2]	[0.7, 0.4]	[0.6, 0.5]	[0.2, 0.3]	[0.5, 0.6]

El decoder genera la traducción palabra por palabra, comenzando con un token especial [START]. En cada paso, analiza el contexto generado por el encoder y busca en su vocabulario español las palabras más probables. Por ejemplo, para la primera palabra, encuentra tres candidatos principales: "El" (85%), "Un" (10%) y "La" (5%), seleccionando "El" por tener la mayor probabilidad. Este proceso continúa hasta completar la traducción "El gato persigue al ratón", donde cada palabra se ha seleccionado considerando tanto el contexto general como las palabras ya traducidas.



Como podés ver, los mecanismos de atención y auto-atención representan un avance revolucionario en el procesamiento del lenguaje natural. Mientras que la atención permite al modelo enfocarse selectivamente en diferentes partes de una entrada para generar cada salida, la auto-atención va un paso más allá al permitir que cada elemento de una secuencia interactúe directamente con todos los demás elementos, creando una rica red de relaciones contextuales. Esta capacidad de procesar el lenguaje de manera no secuencial y establecer conexiones de largo alcance es lo que permitió a los Transformers alcanzar niveles sin precedentes en la comprensión y generación de lenguaje natural.

Capas del Transformer: Las capas de transformer funcionan como un sistema de procesamiento que permite al modelo entender y generar texto. Podemos explicarlo con una analogía: imaginá un equipo de traductores expertos (*encoders*) que primero leen y analizan un texto en el idioma original, y otro equipo (*decoders*) que genera la traducción al nuevo idioma. Cada equipo tiene múltiples personas (*Multi-Head Attention*[82]) que se especializan en diferentes aspectos del lenguaje - algunos se concentran en el contexto,

[82] Este mecanismo permite al modelo enfocarse en diferentes partes de la secuencia de entrada desde múltiples perspectivas, mejorando su capacidad para capturar relaciones complejas.

otros en la gramática, otros en el estilo - y todos trabajan en paralelo. Las capas *feed-forward* son como sesiones de revisión donde toda esta información se procesa y refina. Por ejemplo, al traducir la frase "The cat sat on the mat", los encoders analizarían cada palabra y su relación con las demás, mientras que los decoders tomarían esta información procesada para generar una traducción coherente como "El gato se sentó en la alfombra", considerando tanto el significado literal como el contexto cultural y gramatical.



Positional Encoding: La codificación posicional es una característica crucial de los transformers que les permite entender el orden de las palabras en una frase. A diferencia de los humanos, que leemos naturalmente de izquierda a derecha (o de derecha a izquierda en algunos idiomas), los transformers inicialmente ven todas las palabras al mismo tiempo, como si fueran un conjunto desordenado. Para solucionar esto, cada palabra recibe una "etiqueta de posición" única que indica exactamente dónde está en la frase. Por ejemplo, en la frase "El perro persigue al gato", el orden es crucial para entender quién persigue a quién. Sin codificación posicional, "El gato persigue al perro" significaría exactamente lo mismo para el modelo, lo cual sería un error. Por eso, cada palabra recibe una marca matemática especial que codifica su posición: "El" recibe una marca que indica que es la primera palabra, "perro" que es la segunda, y así sucesivamente. Esto permite al transformer mantener el significado correcto de la frase preservando el orden de las palabras, incluso cuando las procesa todas simultáneamente.

Entrenamiento: El entrenamiento de los transformers requiere grandes cantidades de datos de texto, típicamente recopilados de diversas fuentes de internet como páginas web, libros digitalizados, artículos, documentación técnica y contenido generado por usuarios. Este proceso, conocido como pre-entrenamiento, expone al modelo a billones de palabras y frases en múltiples idiomas y contextos. La calidad y diversidad de estos datos son cruciales: el modelo necesita ver ejemplos de diferentes estilos de escritura, temas, registros lingüísticos y estructuras gramaticales para desarrollar una comprensión robusta del lenguaje. Sin embargo, no todos los datos son igualmente útiles. Los datos de entrenamiento deben pasar por un proceso riguroso de limpieza y filtrado para eliminar contenido duplicado, spam, texto mal formado o inapropiado. Además, es importante que los datos representen un balance adecuado de diferentes dominios y tipos de contenido. Por ejemplo, si el modelo se entrena principalmente con contenido

técnico y formal, podría tener dificultades para entender o generar lenguaje coloquial. Por eso, las empresas que desarrollan transformers invierten considerables recursos en la curación y preparación de sus conjuntos de datos de entrenamiento, buscando un equilibrio entre cantidad y calidad que permita al modelo desarrollar capacidades lingüísticas generales y versátiles.



Modelos de Lenguaje de Gran Escala

BERT (2018)

BERT es un modelo de lenguaje bidireccional que utiliza técnicas de **atención** para predecir palabras enmascaradas en un texto. Esto lo hace útil en tareas como **clasificación de texto**, **análisis de sentimientos** y **respuesta a preguntas**.

GPT (2018)

GPT es un modelo de lenguaje autoregresivo que genera texto palabra por palabra, permitiendo crear contenido coherente y fluido en diversos estilos y géneros. Es ampliamente utilizado para tareas de **generación de texto** como **escritura creativa**, **resúmenes** y **diálogos**.

T5 (2019)

T5 es un modelo versátil que convierte una amplia variedad de tareas de **procesamiento de lenguaje natural**, como **respuesta a preguntas**, **resumen** y **traducción**, en problemas de **traducción texto a texto**.

Impacto: Los transformers representan un avance revolucionario que hizo posible el desarrollo de los Modelos de Lenguaje de Gran Escala (LLMs). Su arquitectura única permite procesar textos mucho más largos y establecer conexiones más complejas entre palabras distantes en una secuencia, superando las limitaciones de los modelos anteriores que perdían información al procesar textos extensos. Esta capacidad, combinada con la habilidad de procesar datos en paralelo (en lugar de secuencialmente), permite entrenar modelos cada vez más grandes con cantidades masivas de datos.

LLMs: Modelos de lenguaje de gran escala

Los modelos basados en la arquitectura Transformer revolucionaron el procesamiento del lenguaje natural, destacando varios hitos que han definido el desarrollo del campo. BERT estableció las bases con su innovadora capacidad de entender el contexto bidireccional, mientras que la familia GPT se convirtió en el estándar para la generación de texto, alcanzando capacidades multimodales sin precedentes. A estos se sumaron modelos como Claude, que se destacó por su razonamiento estructurado y ético, LLaMA, que democratizó el acceso a LLMs a través del código abierto, Gemini, que redefinió la integración multimodal, y

DeepSeek, que introdujo innovaciones en eficiencia computacional. Cada uno de estos modelos aportó características únicas que han expandido las fronteras del procesamiento del lenguaje natural, desde la comprensión contextual hasta la generación de contenido multimodal.

BERT (Bidirectional Encoder Representations from Transformers): Como vimos al comienzo del libro, BERT es un modelo de lenguaje preentrenado bidireccional que se entrena para predecir palabras enmascaradas en una secuencia. Su arquitectura permite capturar el contexto tanto hacia adelante como hacia atrás, mejorando la comprensión del significado de las palabras en diferentes contextos. BERT se utiliza en tareas como la clasificación de texto, el análisis de sentimientos y la respuesta a preguntas.

GPT (Generative Pre-trained Transformer): La familia de modelos GPT ha evolucionado significativamente desde su introducción, logrando avances notables en generación de texto y comprensión del lenguaje natural. GPT-3 marcó un hito con sus 175 mil millones de parámetros, mientras que GPT-3.5 mejoró estas capacidades, siendo la base de ChatGPT. GPT-4 introdujo una arquitectura multimodal capaz de procesar texto e imágenes para generar respuestas más precisas y contextualizadas. En mayo de 2024, OpenAI lanzó GPT-4o, una versión gratuita que amplió las capacidades multimodales al incluir procesamiento de sonidos. En septiembre de 2024, OpenAI presentó la serie o1, incluyendo o1-preview y o1-mini, diseñados para razonar y resolver problemas complejos en áreas STEM. o1-preview destaca por su capacidad de "pensar" antes de responder, logrando un rendimiento excepcional en tareas que requieren pensamiento analítico profundo, como resolver el 83% de los problemas en la Olimpiada Internacional de Matemáticas. Al momento de revisar este párrafo ya tenemos disponible dos modelos nuevos: o3-mini y o3-mini-high.

Claude: Claude, desarrollado por Anthropic, se ha destacado desde su introducción por su capacidad para seguir instrucciones complejas y razonar de manera estructurada. Inicialmente diseñado con técnicas avanzadas de alineamiento para mantener respuestas éticas y precisas, ha sido especialmente útil en tareas que requieren análisis detallado y razonamiento paso a paso. En junio de 2024, Anthropic lanzó Claude 3.5 Sonnet, que mejoró significativamente en tareas de codificación y uso de herramientas. Más tarde, en octubre de 2024, se presentó una versión mejorada de Claude 3.5 Sonnet junto con Claude 3.5 Haiku, optimizado para mayor velocidad y eficiencia. Entre las innovaciones más recientes destaca la función experimental de "computer use", que permite a Claude interactuar con interfaces de usuario al emular acciones humanas, como mover el cursor y hacer clic en botones, ampliando aún más su utilidad en escenarios prácticos y complejos. Estos desarrollos consolidan a Claude como una herramienta avanzada en el ámbito de la inteligencia artificial.

LLaMA/LLaMA 2: LLaMA, desarrollado por Meta, representa un avance significativo en modelos de código abierto desde su lanzamiento inicial en febrero de 2023. La primera versión incluyó modelos de 7B, 13B, 32.5B y 65.2B parámetros, demostrando que era posible lograr un rendimiento competitivo con una arquitectura más eficiente. En julio de 2023, Meta presentó LLaMA 2 en colaboración con Microsoft, disponible en tamaños de 7B, 13B y 70B parámetros, y liberado como código abierto para investigación y uso comercial, lo que catalizó la innovación en la comunidad de IA. En abril de 2024, se lanzó LLaMA 3, que introdujo modelos de 8B y 70.6B parámetros, mejorando la eficiencia y capacidad de procesamiento. Finalmente, en julio de 2024, Meta presentó LLaMA 3.1, un modelo de 405B parámetros, el LLM de código abierto más grande hasta la fecha, con una longitud de contexto ampliada a 128K y soporte para ocho idiomas. Estos desarrollos reflejan el compromiso de Meta con la inteligencia artificial de código abierto, fomentando la innovación y la adaptabilidad en una amplia variedad de aplicaciones.

Gemini: Gemini, desarrollado por Google y lanzado en diciembre de 2023, representa un nuevo paradigma en modelos de inteligencia artificial multimodal. Diseñado desde el principio para comprender y generar texto, código, imágenes y video de manera integrada, Gemini se distingue de otros modelos que añadieron capacidades multimodales posteriormente. Está disponible en tres versiones: Ultra, Pro y Nano, cada una optimizada para diferentes casos de uso y requisitos computacionales. Gemini Nano, por ejemplo, está diseñado para ejecutarse en dispositivos móviles, proporcionando soluciones de inteligencia artificial

accesibles y eficientes. Estas características han posicionado a Gemini como una herramienta versátil y avanzada en el ámbito de la inteligencia artificial, ofreciendo una interacción más fluida y eficiente con diferentes tipos de datos y fomentando su aplicación en una amplia gama de escenarios.

DeepSeek: DeepSeek-R1 es un modelo de inteligencia artificial desarrollado por la startup china DeepSeek, que irrumpió en el panorama tecnológico al ofrecer un rendimiento comparable o superior a modelos líderes como OpenAI o1, pero con una eficiencia de costos y energía significativamente mejorada (presuntamente[83]). Su arquitectura optimizada permite procesar grandes volúmenes de datos con menos recursos computacionales, y su enfoque en el razonamiento lógico y matemático lo hace especialmente apto para tareas que requieren análisis detallado y soluciones paso a paso. Lo que hace especial a DeepSeek es su arquitectura Mixture-of-Experts (MoE), que le permite ser muy eficiente: de sus 671 mil millones de parámetros totales, solo activa 37 mil millones en cada operación. Además, al ser de código abierto, DeepSeek-R1 fomenta la colaboración y la innovación en la comunidad de inteligencia artificial, permitiendo a desarrolladores y empresas utilizar, modificar y mejorar el modelo sin restricciones de licencias propietarias.

Aplicaciones de los LLM



Traducción

Traducción automática de alta calidad entre cientos de idiomas, abriendo nuevas posibilidades de comunicación global.



Asistentes Virtuales

Chatbots y asistentes de voz con capacidades de interacción natural, anticipando y respondiendo a las necesidades del usuario.



Generación de Contenido

Creación automática de artículos, historias, poesía y otros formatos de texto, inspirando y complementando la producción humana.



Análisis de Textos

Resumen de contenidos, detección de sentimientos y extracción de insights clave a partir de grandes volúmenes de información.



Codificación y Programación Asistida

Generación y análisis automático de código, acelerando el desarrollo de software y reduciendo errores.



Automatización de Procesos Empresariales

Automatización inteligente de tareas repetitivas y flujos de trabajo administrativos, aumentando la eficiencia operativa.

Los Modelos de Lenguaje de Gran Escala (LLMs) transformaron la manera en que interactuamos con la tecnología, ofreciendo un amplio abanico de aplicaciones prácticas. Estas herramientas revolucionaron campos tan diversos como la creación de contenido, donde pueden generar textos creativos; la atención al

[83] Escribo estas palabras el 2 de Febrero de 2025 en una de mis últimas revisiones.

cliente, a través de chatbots cada vez más sofisticados; y el análisis de datos, con capacidades para resumir documentos extensos. Su impacto se extiende más allá del procesamiento de lenguaje tradicional, convirtiéndose en herramientas fundamentales para la toma de decisiones empresariales y la automatización de tareas complejas.

📝 **Generación de contenido**: Los LLMs son ampliamente empleados en la creación de contenido automatizado, que va desde artículos, publicaciones en redes sociales, guiones y resúmenes. Son especialmente útiles para acelerar procesos creativos o de marketing. *Ejemplo: "Escribe un tweet promocional sobre nuestro nuevo café orgánico" → "☕ ¡Descubre el sabor de la naturaleza! Nuestro nuevo café orgánico, cultivado en las alturas de Colombia, te espera. #CaféOrgánico #SaborNatural"*

🤖 **Chatbots y asistentes virtuales**: Una de las aplicaciones más comunes es la creación de chatbots de atención al cliente que responden consultas y generan recomendaciones en tiempo real. Estos asistentes son capaces de mejorar la interacción con los clientes y optimizar el proceso de ventas. *Ejemplo: "Cliente: ¿Cuál es su política de devoluciones?" → "Bot: Las devoluciones son gratuitas dentro de los primeros 30 días de compra. ¿Necesita iniciar una devolución?"*

😊 **Análisis de sentimientos**: Los LLMs se utilizan para analizar emociones y sentimientos en textos, permitiendo a las empresas medir la opinión pública sobre sus productos o servicios a través de redes sociales y reseñas de clientes. *Ejemplo: "Este producto superó todas mis expectativas, ¡increíble servicio!" → Sentimiento: Muy Positivo (0.92)*

🌍 **Traducción de idiomas**: Los LLMs permiten traducir textos entre distintos idiomas con alta precisión, facilitando el acceso a información en múltiples lenguas, lo que es crucial en un entorno globalizado. *Ejemplo: "Hello, how are you doing today?" → "¡Hola!, ¿cómo estás hoy?" → "Bonjour, comment allez-vous aujourd'hui?"*

👨‍💻 **Desarrollo de software**: Estos modelos también pueden generar y depurar código, lo que facilita a los programadores acelerar ciertas tareas básicas y optimizar el desarrollo de aplicaciones. *Ejemplo: "Crea una función para calcular el factorial" → python def factorial(n): return 1 if n <= 1 else n * factorial(n-1)*

📚 **Resúmenes y clasificaciones de texto**: Los LLMs ayudan a sintetizar información de documentos extensos, lo cual es útil para profesionales que necesitan condensar grandes volúmenes de datos rápidamente. *Ejemplo: [Artículo de 1000 palabras sobre cambio climático] → "Resumen: El estudio revela un aumento de 1.5°C en temperaturas globales, destacando la urgencia de reducir emisiones de CO2 y adoptar energías renovables."*

Ventajas de los Transformers y LLMs

Comprensión contextual: Los transformers capturan el contexto completo de una secuencia, mejorando la precisión y la relevancia de las respuestas generadas. Por ejemplo, en la frase "El banco está cerrado por remodelación", el modelo entiende que "banco" se refiere a una institución financiera y no a un asiento, gracias a su capacidad de analizar simultáneamente todas las palabras de la oración y sus relaciones entre sí. Esta comprensión profunda del contexto permite generar respuestas más coherentes y apropiadas.

Versatilidad: Pueden ser aplicados a una amplia gama de tareas de NLP, desde la traducción automática hasta la generación de contenido y el análisis de sentimientos. Un mismo modelo transformer puede ser ajustado para realizar múltiples tareas diferentes. Por ejemplo, GPT-4 puede escribir código, analizar textos, traducir idiomas, y hasta resolver problemas matemáticos complejos, todo con la misma arquitectura base. Esta flexibilidad reduce la necesidad de desarrollar modelos específicos para cada tarea.

Capacidad de escalado: Los LLMs pueden escalarse para manejar grandes volúmenes de datos y aprovechar eficientemente la potencia computacional moderna, como las GPUs y TPUs. Esta capacidad de escalado ha permitido la creación de modelos cada vez más potentes, desde los primeros transformers con millones de parámetros hasta modelos actuales como GPT-4 con billones de parámetros. El aumento en tamaño y capacidad computacional se traduce directamente en mejores resultados y mayor comprensión del lenguaje.

Transferencia de conocimientos: Los modelos preentrenados pueden adaptarse rápidamente a nuevas tareas con una cantidad relativamente pequeña de datos adicionales, facilitando el desarrollo de nuevas aplicaciones. Por ejemplo, un modelo entrenado en inglés general puede ser ajustado para entender jerga médica o legal con relativamente pocos ejemplos específicos del dominio. Esta capacidad, conocida como "fine-tuning", ahorra tiempo y recursos significativos en el desarrollo de aplicaciones especializadas, ya que no es necesario entrenar cada nuevo modelo desde cero.

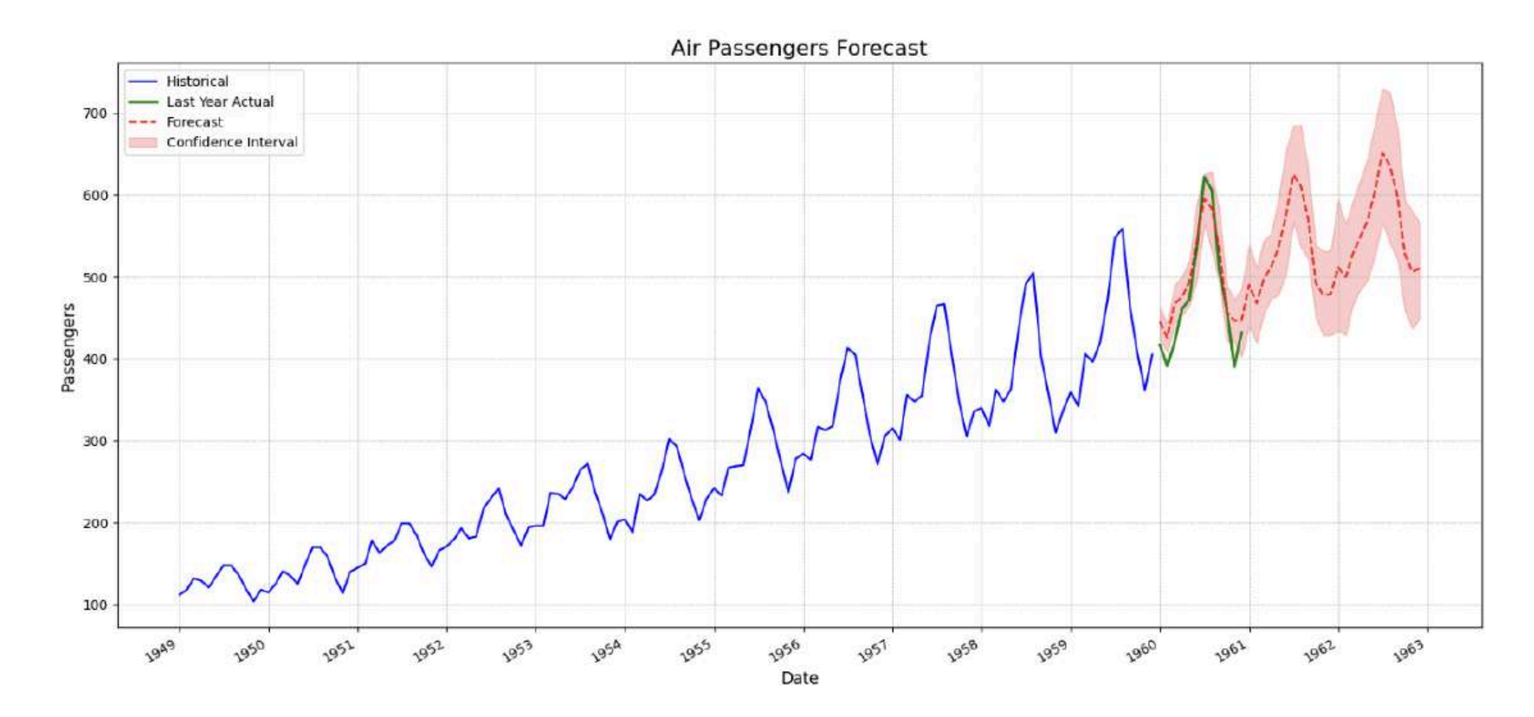
Desafíos de los Transformers y LLMs

Requerimientos computacionales: Entrenar y desplegar modelos de lenguaje de gran escala es computacionalmente intensivo y costoso, requiriendo hardware especializado y grandes cantidades de energía. Por ejemplo, se estima que entrenar un modelo como GPT-4 puede costar varios millones de dólares en infraestructura computacional y consumir tanta energía como varios miles de hogares en un año. Esto limita significativamente quién puede desarrollar y mantener estos modelos, centralizando el poder en grandes corporaciones tecnológicas.

Tamaño del modelo: Los LLMs pueden ser extremadamente grandes, lo que presenta desafíos en términos de almacenamiento, implementación y eficiencia de ejecución. Por ejemplo, mientras que un modelo pequeño de procesamiento de texto puede ocupar unos pocos megabytes, modelos como GPT-4 requieren cientos de gigabytes solo para almacenar sus parámetros. Esto hace que sea difícil ejecutar estos modelos en dispositivos comunes como teléfonos móviles o computadoras personales, requiriendo en su lugar grandes centros de datos y servicios en la nube.

Sesgo y equidad: Los modelos de lenguaje de gran escala pueden aprender sesgos presentes en los datos de entrenamiento, lo que puede llevar a resultados discriminatorios o injustos. Por ejemplo, si los datos de entrenamiento contienen más ejemplos de hombres en roles de liderazgo que de mujeres, el modelo podría tender a asociar posiciones de poder con el género masculino. Estos sesgos pueden manifestarse de formas sutiles pero significativas, como generar diferentes tipos de respuestas según el género, la etnia o la nacionalidad mencionada en la consulta.

Privacidad y seguridad: Los LLMs pueden generar información sensible o confidencial, planteando preocupaciones sobre la privacidad y la seguridad de los datos. Por ejemplo, si un modelo se entrena con datos que incluyen información personal o propietaria, existe el riesgo de que pueda revelar inadvertidamente esta información en sus respuestas. Además, estos modelos pueden ser vulnerables a ataques adversarios que busquen extraer información sensible o manipular sus respuestas. La necesidad de equilibrar la utilidad del modelo con la protección de la privacidad representa un desafío continuo en el campo.



Aunque los Transformers revolucionaron el **procesamiento del lenguaje natural**, su arquitectura fundamental resulta ser sorprendentemente versátil. La clave de esta versatilidad está en su capacidad para procesar cualquier tipo de secuencia, no solo texto.

Esto para mí fue una revelación importantísima ya que durante la mayor parte de mi carrera profesional me he encontrado con el problema de la predicción futura de variables económicas o financieras trabajando en diferentes empresas o para diferentes clientes. Esto es, el clásico problema del *forecasting*.

Resolver este problema de forma eficiente y a gran escala es sumamente complejo ya que los métodos que han tenido más éxito históricamente son modelos estadísticos clásicos[84] y la inteligencia artificial parecería aún no haber alcanzado buenos resultados en este nicho. Pero como dijimos, aparece la arquitectura del Transformer con su asombrosa capacidad de hacer predicciones secuenciales (*tokens* o *palabras)* teniendo en cuenta el contexto y las dependencias de largo plazo y entonces comenzaron a surgir ideas de cómo esto se podría adaptar al problema clásico de *forecasting* de series temporales. Al fin y al cabo, ¿qué es una serie temporal sino una secuencia de valores que evolucionan en el tiempo? Si un

[84] Los modelos estadísticos clásicos para series temporales incluyen principalmente: ARIMA (Autoregressive Integrated Moving Average) y sus variantes como SARIMA para series estacionales, que modelan la serie como una combinación de componentes autorregresivos y de media móvil; los métodos de Suavizado Exponencial como Holt-Winters, que ponderan las observaciones dando más peso a las más recientes; y los modelos estructurales como STL (Seasonal and Trend decomposition using Loess) que descomponen la serie en sus componentes de tendencia, estacionalidad y residuos. Estos modelos han dominado el campo durante décadas por su interpretabilidad y buenos resultados en series con patrones claros, aunque suelen requerir ajuste manual de parámetros y tienen dificultades con patrones no lineales complejos.

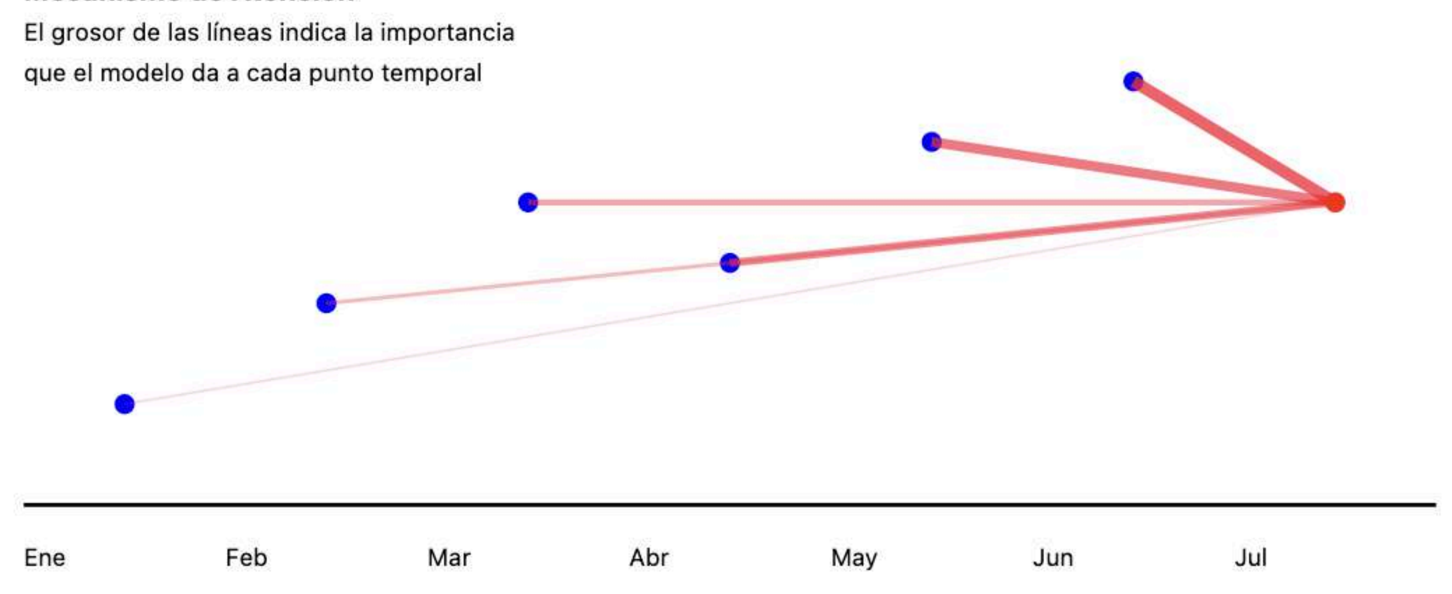
Transformer puede aprender que en una oración "el gato" probablemente va seguido de "duerme", ¿por qué no podría aprender que las *ventas de diciembre* suelen ser más altas que las de *noviembre*? O que después de tres meses de crecimiento sostenido, suele haber una corrección a la baja?

Paralelismo entre Lenguaje Natural y Series Temporales

Aspecto	Lenguaje Natural	Series Temporales
Unidad básica	Palabras o tokens	Valores numéricos en puntos temporales
Predicción	el gato → duerme	ventas noviembre → ventas diciembre
Contexto	Palabras anteriores en la oración	Valores anteriores en la serie
Patrones	Estructuras gramaticales, frases comunes	Estacionalidad, tendencias, ciclos
Dependencias	Referencias entre palabras distantes	Correlaciones entre períodos distantes
Atención	Enfoque en palabras relevantes para la siguiente predicción	Enfoque en períodos relevantes para el siguiente valor

El mecanismo de *atención*, que permite a un Transformer determinar qué partes de una secuencia son relevantes para predecir la siguiente palabra, puede adaptarse para identificar qué puntos temporales pasados son más importantes para predecir el siguiente valor. Por ejemplo, en ventas minoristas, el modelo podría "prestar atención" automáticamente a los mismos meses del año anterior para capturar patrones estacionales.

Así como GPT fue entrenado con miles de millones de textos para comprender y generar lenguaje, por qué no pensar en un modelo Transformer pre-entrenado con millones de series temporales que pudiera generalizar y aprender patrones universales en datos secuenciales. La visión era ambiciosa pero lógica: si un modelo puede aprender patrones generales en el lenguaje, ¿por qué no podría aprender patrones generales en series temporales, independientemente de si representan ventas, demanda, tráfico web o cualquier otra métrica que evolucione en el tiempo?



Lógicamente comenzaron a surgir investigaciones y modelos concretos[85], y en mi propia investigación[86], desarrollé una adaptación específica del Transformer para resolver uno de los mayores desafíos en predicción empresarial: trabajar con datos históricos limitados[87]. La solución vino en forma de un sistema dual de Transformers, cada uno con un propósito específico pero complementario.

El primer Transformer se especializa en predicción puntual. Imaginá un gerente de ventas experimentado que, tras analizar el historial de ventas, los patrones estacionales y las tendencias, puede estimar las ventas del próximo mes. De manera similar, este modelo examina los últimos 60 meses de datos y predice el valor del siguiente mes.

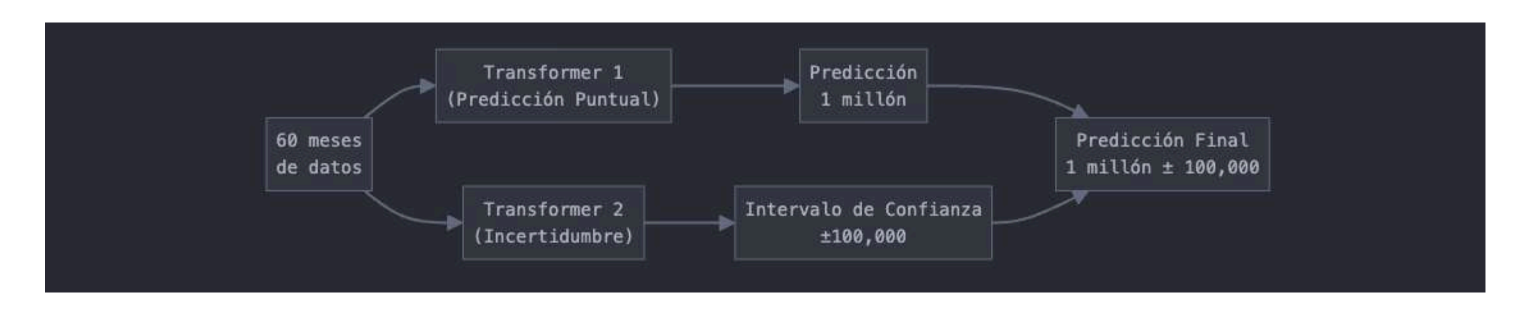
---

[85] La evolución de modelos de deep learning para series temporales ha sido notable en los últimos años. Los primeros intentos adaptaron arquitecturas RNN y CNN, destacando modelos como DeepAR y ESRNN. Un avance significativo llegó con N-BEATS y NHITS (Oreshkin et al., 2019, 2021), que introdujeron nuevas técnicas de interpretabilidad y resolución temporal. La adaptación de Transformers a series temporales produjo modelos como TFT, Autoformer y Fedformer (Zhou et al., 2021; Smith et al., 2021), culminando con TimeGPT (Garza y Mergenthaler-Canseco, 2022), que demostró el potencial de los modelos pre-entrenados en este campo.

[86] A día de hoy no cuento con ninguna publicación sobre esta investigación, sino que lo he hecho dentro del ámbito de mi empresa y aún no hemos tomado ninguna decisión al respecto. De todas formas me vale muy bien a fines didácticos y no creo que sea demasiado importante el detalle de la investigación para el lector promedio.

[87] En la práctica empresarial, es extremadamente raro encontrar series temporales con más de 2-3 años de historia consistente. Esto se debe a múltiples factores: (1) Cambios en sistemas informáticos o migraciones que resultan en pérdida o inconsistencia de datos históricos; (2) Renovación frecuente del catálogo de productos, donde los nuevos productos no tienen histórico y los antiguos dejan de ser relevantes; (3) Reorganizaciones en las jerarquías de productos o regiones que hacen incomparables los datos antiguos con los actuales; (4) Cambios en la metodología de registro de datos o en las definiciones de métricas clave; (5) Fusiones y adquisiciones que resultan en sistemas y datos no compatibles; y (6) Políticas de retención de datos que limitan el almacenamiento histórico. Esta realidad operativa hace que los modelos que requieren largos historiales de datos sean poco prácticos en entornos empresariales reales.

El segundo Transformer añade un elemento crucial: la incertidumbre. Es como si además de la predicción, nuestro gerente pudiera cuantificar su nivel de confianza basado en la volatilidad histórica y los factores de riesgo. Este modelo no solo dice "las ventas serán de 1 millón", sino que añade "con un 95% de confianza estarán entre 900,000 y 1,100,000". Esta información adicional es invaluable para la planificación de inventario y la gestión de recursos.





Los resultados obtenidos con el conjunto de datos M4[88], que comprende 48,000 series temporales mensuales de diversos dominios, fueron notablemente positivos. El modelo alcanzó un sMAPE[89] global de 13.43, con un desempeño particularmente destacado en series demográficas. Lo más significativo es que estos resultados son competitivos con modelos estado del arte que requieren historiales mucho más extensos, mientras que mi propuesta logra estos resultados con solo 5 años de historia. Además, el modelo demostró una capacidad robusta para predicciones "zero-shot" en series temporales nunca antes vistas, lo que sugiere que logra capturar patrones fundamentales en la evolución temporal de las variables. La incorporación del componente probabilístico permitió no solo realizar predicciones puntuales precisas, sino también cuantificar la incertidumbre asociada a cada predicción, un aspecto crucial para la toma de decisiones en entornos empresariales.

Finalmente, y en mi opinión, la aplicación de Transformers al forecasting de series temporales marca un punto de inflexión en este campo. Su capacidad para adaptarse exitosamente más allá del procesamiento del lenguaje natural representa solo el comienzo

[88] M4 es el conjunto de datos de la cuarta competición Makridakis, que contiene 100,000 series temporales de diferentes dominios como finanzas, economía, demografía e industria. Esta competición, realizada en 2018, se ha convertido en el benchmark más importante para evaluar la precisión de modelos de predicción de series temporales, reuniendo a investigadores y profesionales de todo el mundo para comparar sus soluciones bajo las mismas condiciones.

[89] sMAPE (Symmetric Mean Absolute Percentage Error) es una métrica que mide el error de predicción en términos porcentuales, variando entre 0% y 100%. A diferencia del MAPE tradicional, el sMAPE es simétrico, lo que significa que penaliza igualmente las sobreestimaciones y subestimaciones, haciéndolo más adecuado para evaluar predicciones de series temporales. Un sMAPE de 13.43 indica que, en promedio, las predicciones se desvían un 13.43% de los valores reales.

de una revolución en la predicción de series temporales. Su evolución en los próximos años promete dominar este campo por tres razones fundamentales:

1. Capacidad excepcional para capturar patrones complejos: Los Transformers no solo identifican tendencias simples, sino que pueden detectar interrelaciones sutiles y dependencias de largo plazo en los datos, algo crucial en series temporales empresariales.
2. Adaptabilidad a escenarios con datos limitados: Su arquitectura permite obtener resultados robustos incluso con historiales reducidos de 5 años o menos, una restricción común en entornos empresariales reales que otros modelos no manejan eficientemente.
3. Viabilidad práctica en entornos productivos: La posibilidad de utilizar un único modelo pre-entrenado, en lugar de mantener y actualizar cientos de modelos individuales, representa una ventaja operativa significativa en términos de mantenimiento, recursos computacionales y consistencia en las predicciones.

Esta convergencia de capacidades técnicas y practicidad operativa sugiere que los Transformers no solo mejorarán la precisión de nuestras predicciones, sino que también transformarán fundamentalmente cómo las organizaciones abordan sus necesidades de forecasting, permitiendo soluciones más escalables, mantenibles y precisas.

12. 🥷 Agentes IA (Agentive AI)



IA Basada en Agentes: El Futuro de la Inteligencia Artificial

La IA basada en agentes es una rama avanzada que desarrolla sistemas autónomos. Estos sistemas toman decisiones y actúan de forma independiente para lograr objetivos específicos.

A diferencia de la IA tradicional, estos agentes son proactivos y adaptativos.

Si los Transformers nos permiten soñar con una inteligencia artificial que genere una revolución tecnológica en el sentido que la hemos descrito anteriormente, tendrá que ser por que nos permita mucho más que *chatear*, muchísimo más que solamente darnos respuestas a preguntas o generar textos interesantes. Una verdadera IA impactante en el mundo es una IA que puede **interactuar** con él y afectarlo positivamente. Una IA que entiende nuestros objetivos y nos asiste en la toma de decisiones. Esa es la clase de IA que verdaderamente impactará en el mundo y su máxima expresión actualmente está en pleno desarrollo y la conocemos como **IA basada en Agentes**.

La inteligencia artificial basada en agentes (Agentive AI) es una rama avanzada de la inteligencia artificial que se enfoca en el desarrollo de sistemas autónomos que pueden tomar decisiones y realizar acciones para cumplir objetivos específicos de manera proactiva y autónoma. A diferencia de las IA tradicionales que ejecutan tareas específicas cuando se les ordena, la IA basada en agentes se centra en sistemas que actúan de manera independiente, adaptándose y aprendiendo del entorno para optimizar sus acciones y resultados.

Fundamentos de la IA Basada en Agentes

Agente

Entidad autónoma que percibe y actúa en su entorno para alcanzar objetivos.

Entorno

Todo lo que rodea al agente, puede ser dinámico y parcialmente observable.

Sensores y Actuadores

Permiten al agente percibir el entorno y actuar sobre él.

Política y Función de Valor

Definen las estrategias de toma de decisiones y evaluación de estados.



Fundamentos de la Agentive AI

🤖 **Agente**: Un agente es una entidad autónoma que percibe su entorno y actúa sobre él para alcanzar sus objetivos. Los agentes pueden ser físicos, como robots, o virtuales, como software inteligente. En general nos vamos a referir a agentes virtuales, pero el término proviene del campo del aprendizaje por refuerzo que hemos visto anteriormente.

🌍 **Entorno**: El entorno es todo lo que rodea al agente y con lo que interactúa. Puede ser dinámico, estocástico y parcialmente observable, lo que significa que el agente debe adaptarse continuamente a cambios e incertidumbres. Por ejemplo, para un agente de inversión financiera, su entorno incluiría los precios de las acciones en tiempo real, las noticias económicas, los indicadores del mercado, las tasas de interés y los sentimientos de los inversores. Este entorno es:

- Dinámico: los precios y condiciones del mercado cambian constantemente
- Estocástico: hay elementos aleatorios e impredecibles (como eventos geopolíticos inesperados que afectan al mercado)
- Parcialmente observable: el agente no puede ver toda la información relevante (como las decisiones

futuras de los bancos centrales o los planes de otros inversores) Por lo tanto, el agente debe tomar decisiones de inversión basándose en información incompleta y en constante cambio, similar a como lo haría un trader humano.

👀 **Sensores y actuadores**: Los *sensores* permiten al agente percibir el estado del entorno, mientras que los *actuadores* permiten al agente influir en el entorno. En el caso de nuestro agente de inversión financiera, los sensores serían las interfaces de programación (APIs) que le permiten:

- Monitorear precios de acciones en tiempo real
- Recibir noticias financieras
- Analizar informes económicos
- Seguir indicadores técnicos del mercado
- Detectar sentimientos en redes sociales sobre empresas

Y sus actuadores serían los mecanismos que le permiten:

- Ejecutar órdenes de compra y venta
- Ajustar los porcentajes de la cartera
- Establecer órdenes límite
- Implementar estrategias de cobertura

Por ejemplo, si los sensores detectan una caída significativa en el precio de una acción junto con noticias negativas sobre la empresa, el agente podría utilizar sus actuadores para ejecutar una orden de venta o implementar una estrategia de cobertura, dependiendo del perfil de riesgo del inversor y las condiciones generales del mercado.

📊 **Política**: La política de un agente define el conjunto de reglas o estrategias que utiliza para tomar decisiones y seleccionar acciones en función del estado del entorno. Para entenderlo de manera intuitiva, podemos pensar en la política como el "manual de instrucciones" o el "conjunto de reglas de decisión" que sigue el agente. Al igual que un inversor experimentado tiene estrategias claras sobre cuándo comprar o vender basadas en ciertos indicadores, o un jugador de ajedrez tiene principios que guían sus movimientos según el estado del tablero, la política del agente determina qué acción tomar en cada situación posible. Por ejemplo, en nuestro agente de inversión financiera, una política podría ser "vender si el precio cae más del 5% en una hora y el volumen de ventas es alto" o "aumentar la posición si hay noticias positivas y el análisis técnico muestra una tendencia alcista".

💰 **Función de Valor**: La función de valor estima el valor esperado de estar en un estado determinado, considerando las recompensas futuras posibles. Es crucial para que el agente tome decisiones que maximicen su recompensa a largo plazo. Para entenderlo de manera intuitiva, es similar a cómo un jugador de ajedrez evalúa una posición no solo por la ventaja inmediata, sino también por las oportunidades futuras que ofrece. En nuestro ejemplo del agente de inversión, la función de valor no sólo consideraría el beneficio inmediato de una operación, sino también sus implicaciones futuras: una venta podría generar una pequeña ganancia ahora, pero ¿qué pasa si las perspectivas a largo plazo de la empresa son muy positivas? La función de valor ayudaría al agente a equilibrar las recompensas inmediatas con el potencial de ganancias futuras, similar a cómo un inversor experimentado piensa en términos de "valor presente" de oportunidades futuras.

🧠 **Aprendizaje y Adaptación**: Algunos agentes inteligentes utilizan técnicas de aprendizaje automático, como el aprendizaje por refuerzo, para mejorar continuamente su comportamiento y rendimiento en función de la experiencia adquirida. Siguiendo con nuestro ejemplo del agente de inversión financiera, el aprendizaje funcionaría de manera similar a cómo un trader novato se convierte en experto: a través de la experiencia y la retroalimentación. Si el agente toma una decisión de vender cuando una acción cae un 5% y luego observa que la acción se recupera rápidamente y sube un 15%, aprenderá que quizás ese umbral del 5% era

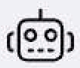
demasiado sensible para ese tipo particular de acción o para esas condiciones de mercado. Con el tiempo, el agente puede ajustar sus umbrales, refinar sus estrategias y desarrollar una comprensión más matizada de los patrones del mercado. Por ejemplo, podría aprender que las caídas en ciertos sectores tecnológicos suelen ser más volátiles pero también más propensas a recuperaciones rápidas, mientras que las caídas en sectores más tradicionales podrían requerir una estrategia de salida más conservadora.

Aplicaciones de la IA Basada en Agentes



Asistentes Personales

Realizan tareas variadas, anticipándose a las necesidades del usuario.



Robótica Autónoma

Robots y drones que operan en entornos dinámicos sin intervención humana.



Gestión Empresarial

Automatizan tareas, optimizan procesos y proporcionan análisis predictivos.



Asistencia en Salud

Monitorean signos vitales y proporcionan alertas en emergencias.

Agente IA de Inversiones Personales: InversorIA



Vamos a comenzar un ejemplo simulando el funcionamiento de un Agente IA ficticio para que puedas ver cómo concebimos actualmente esta tecnología. Te presento entonces a

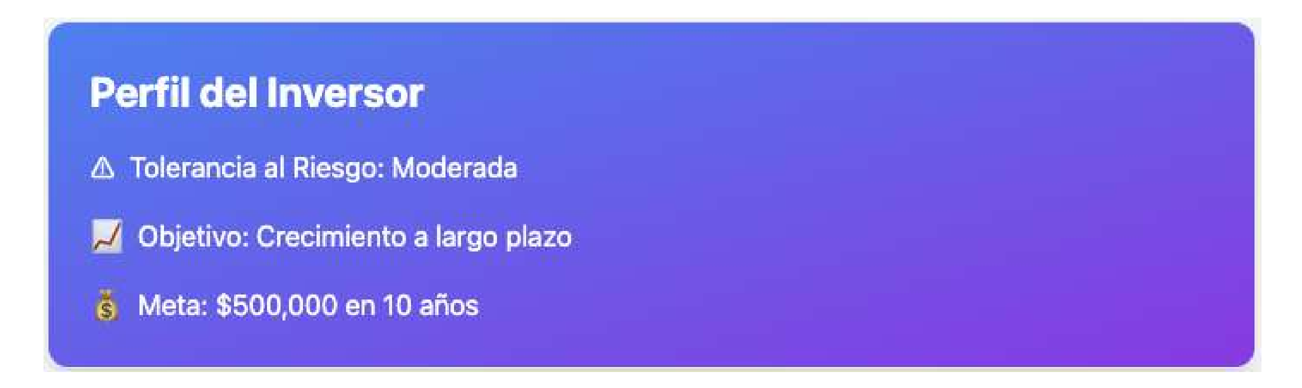
InversorIA, un agente de inteligencia artificial diseñado para optimizar y gestionar inversiones personales. A continuación, te detallo cómo funciona este agente en la práctica:

1. Inicialización y perfil del inversor

Cuando un usuario comienza a utilizar ***InversorIA***, el agente inicia un proceso de recopilación de información:

- **Cuestionario de perfil**: InversorIA presenta una serie de preguntas para evaluar la tolerancia al riesgo, objetivos financieros y restricciones del usuario.
- **Análisis de cuentas existentes**: Con permiso del usuario, el agente conecta y analiza las cuentas financieras existentes para entender la situación financiera actual.
- **Establecimiento de objetivos**: El usuario especifica sus metas financieras, como "ahorrar para la jubilación" o "comprar una casa en 5 años".

InversorIA define este perfil para el usuario:



2. Análisis de Mercado y Datos

InversorIA constantemente recopila y analiza datos de múltiples fuentes:

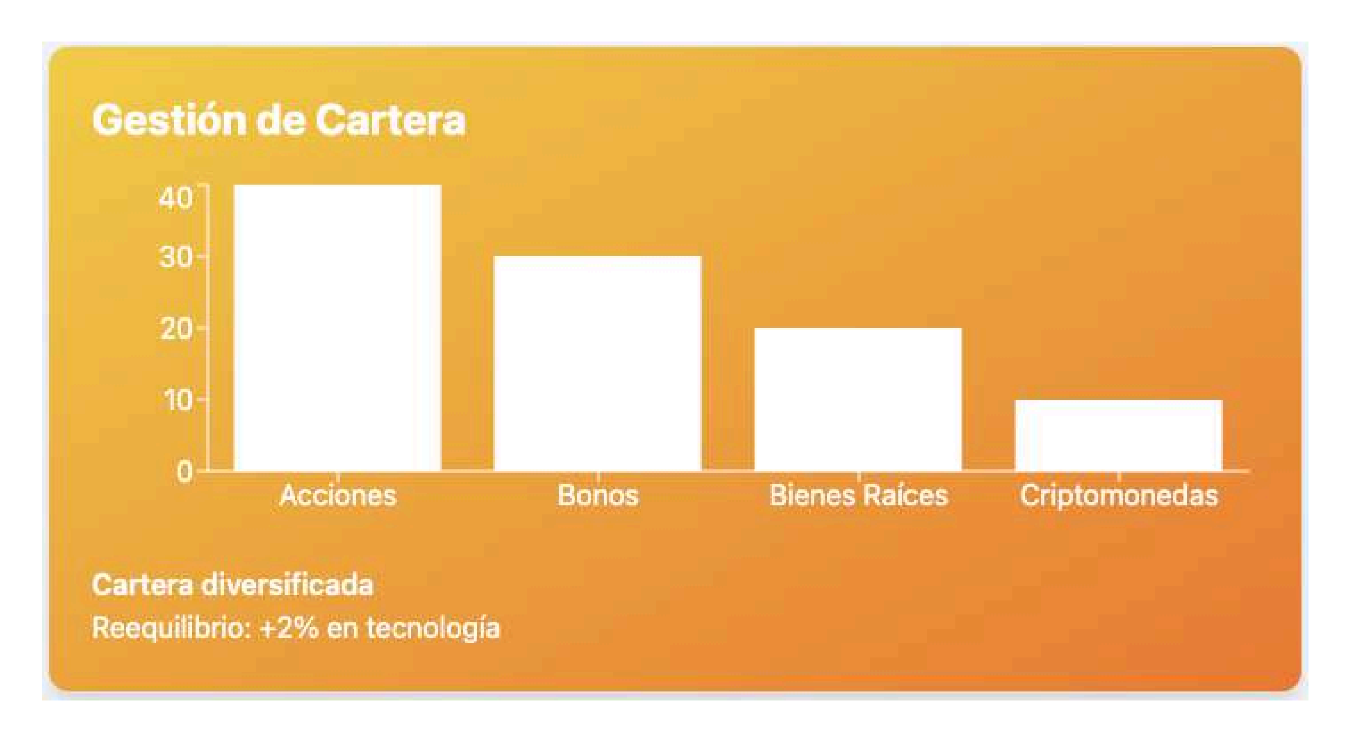


- **Datos de mercado en tiempo real**: Precios de acciones, bonos, materias primas y otros activos.
- **Noticias y reportes financieros**: Utiliza procesamiento de lenguaje natural para analizar noticias económicas, informes corporativos y opiniones de expertos.
- **Indicadores económicos**: Monitorea tasas de interés, inflación, crecimiento económico y otros indicadores macroeconómicos.
- **Análisis técnico**: Aplica algoritmos de deep learning para identificar patrones en los datos históricos de precios.

3. Diseño y Gestión de Cartera

Basándose en el perfil del inversor y el análisis de mercado, ***InversorIA***:

- **Crea una cartera inicial**: Utiliza algoritmos de optimización para diseñar una cartera diversificada que se alinee con los objetivos y la tolerancia al riesgo del usuario.
- **Reequilibra dinámicamente**: Ajusta constantemente las asignaciones de activos para mantener el balance de riesgo-rendimiento deseado.
- **Implementa estrategias de inversión**: Aplica técnicas como inversión por valor, momentum, o estrategias de factores, según sea apropiado para el perfil del usuario.



4. Ejecución de Operaciones

InversorIA ejecuta operaciones de manera autónoma:

- **Algoritmos de ejecución**: Utiliza estrategias de negociación algorítmica para minimizar el impacto en el mercado y obtener los mejores precios.

- **Timing de mercado**: Aunque no intenta "ganarle" constantemente al mercado, aprovecha ineficiencias de corto plazo cuando se presentan oportunidades claras.
- **Gestión de órdenes**: Implementa y gestiona órdenes límite, stop-loss y otros tipos de órdenes avanzadas.



5. Gestión de riesgos

El agente implementa varias capas de gestión de riesgos:



- **Diversificación dinámica**: Ajusta la diversificación de la cartera basándose en las correlaciones cambiantes entre activos.
- **Hedging**: Utiliza instrumentos como opciones o futuros para proteger la cartera contra riesgos específicos cuando es necesario.
- **Monitoreo de riesgos**: Calcula y monitorea constantemente métricas de riesgo como VaR (Valor en Riesgo) y volatilidad.

6. Optimización Fiscal

InversorIA considera las implicaciones fiscales en sus decisiones:

- **Cosecha de pérdidas fiscale**s: Vende estratégicamente activos con pérdidas para compensar ganancias de capital.
- **Ubicación de activos**: Coloca activos de manera óptima entre cuentas gravables y no gravables (como IRAs o 401(k)s).
- **Consideraciones de horizonte temporal**: Favorece estrategias de retención a largo plazo para minimizar impuestos cuando es apropiado.



7. Aprendizaje y Adaptación

El agente mejora continuamente su desempeño:

- **Aprendizaje por refuerzo**: Utiliza el rendimiento histórico de sus decisiones para refinar sus estrategias de inversión.
- **Adaptación a cambios de mercado**: Ajusta sus modelos predictivos basándose en cambios en las dinámicas del mercado.
- **Personalización continua**: Refina su entendimiento de las preferencias del usuario basándose en sus interacciones y feedback.

Aprendizaje y Adaptación

🧠 Precisión del modelo: 92%

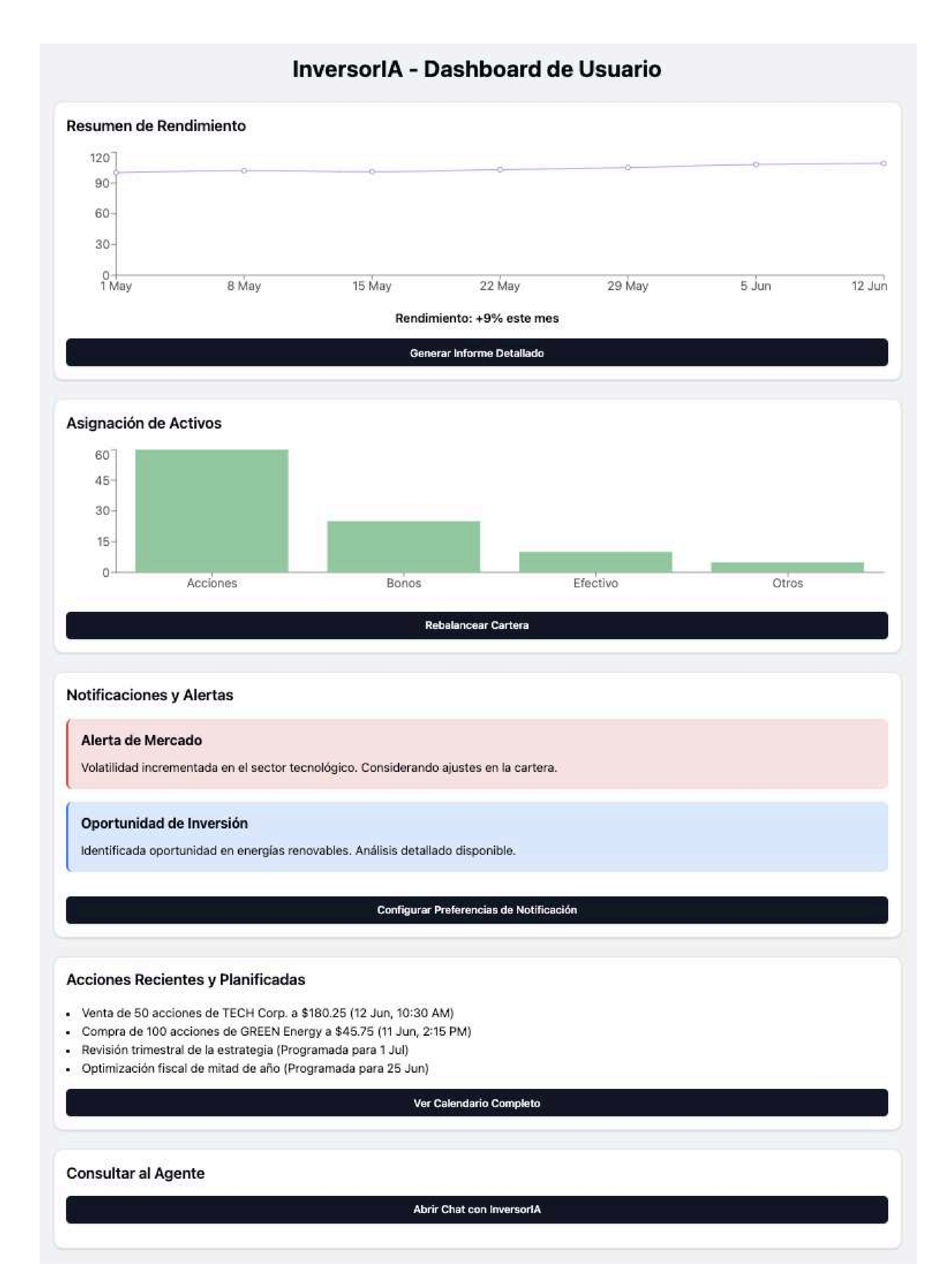
📈 Mejora en rendimiento: +2.5%

⚠ Nueva estrategia: Momentum en Small Caps

8. Comunicación con el Usuario

InversorIA mantiene al usuario informado y en control:

- **Dashboards personalizados**: Proporciona visualizaciones en tiempo real del rendimiento de la cartera y decisiones de inversión.
- **Alertas y notificaciones**: Informa al usuario sobre eventos significativos o cambios importantes en la estrategia.
- **Explicaciones claras**: Utiliza técnicas de IA explicable para proporcionar justificaciones claras de sus decisiones.
- **Control del usuario**: Permite al usuario establecer restricciones, ajustar preferencias o anular decisiones específicas del agente.


InversorIA - Dashboard de Usuario
Resumen de Rendimiento
120
90
60
30
0
1 May
8 May
15 May
22 May
29 May
5 Jun
12 Jun
Rendimiento: +9% este mes
Generar Informe Detallado
Asignación de Activos
60
45
30
15
0
Acciones
Bonos
Efectivo
Otros
Rebalancear Cartera
Notificaciones y Alertas
Alerta de Mercado
Volatilidad incrementada en el sector tecnológico. Considerando ajustes en la cartera.
Oportunidad de Inversión
Identificada oportunidad en energías renovables. Análisis detallado disponible.
Configurar Preferencias de Notificación
Acciones Recientes y Planificadas
Venta de 50 acciones de TECH Corp. a $180.25 (12 Jun, 10:30 AM)
Compra de 100 acciones de GREEN Energy a $45.75 (11 Jun, 2:15 PM)
Revisión trimestral de la estrategia (Programada para 1 Jul)
Optimización fiscal de mitad de año (Programada para 25 Jun)
Ver Calendario Completo
Consultar al Agente
Abrir Chat con InversorIA

Ejemplo de Flujo de Trabajo con InversorIA

InversorIA Chat

🚨 Alerta de Mercado: Se ha detectado una caída significativa en el sector tecnológico debido a noticias negativas. Estoy analizando el impacto en tu cartera.

Análisis completado. La caída tiene un impacto potencial del 3.5% en el valor total de tu cartera. Sin embargo, veo oportunidades para mitigar y potencialmente beneficiarnos de esta situación.

Entiendo. ¿Qué acciones recomiendas tomar?

Basado en mi análisis, recomiendo lo siguiente: 1. Reequilibrar la cartera vendiendo un 2% de nuestras posiciones en sectores de alto rendimiento (energía y salud). 2. Utilizar estos fondos para comprar selectivamente acciones tecnológicas ahora infravaloradas, específicamente en empresas con fuertes fundamentos y potencial de recuperación.

Además, para protegernos contra una mayor volatilidad, sugiero implementar una estrategia de cobertura utilizando opciones put en el índice tecnológico principal.

Suena bien. ¿Puedes proceder con estas acciones?

Ciertamente. Procederé con las acciones recomendadas. Aquí está un resumen de las transacciones que realizaré: - Venta: 100 acciones de EnergyPlus (sector energético) a $75.50 - Venta: 50 acciones de HealthNow (sector salud) a $120.25 - Compra: 80 acciones de TechInnovate a $95.75 - Compra: 60 acciones de DataPioneer a $88.50 - Adquisición de opciones put sobre el índice NASDAQ-100 con strike price 5% por debajo del nivel actual, vencimiento a 3 meses

Gracias por el detalle. ¿Cómo monitorearás el impacto de estas decisiones?

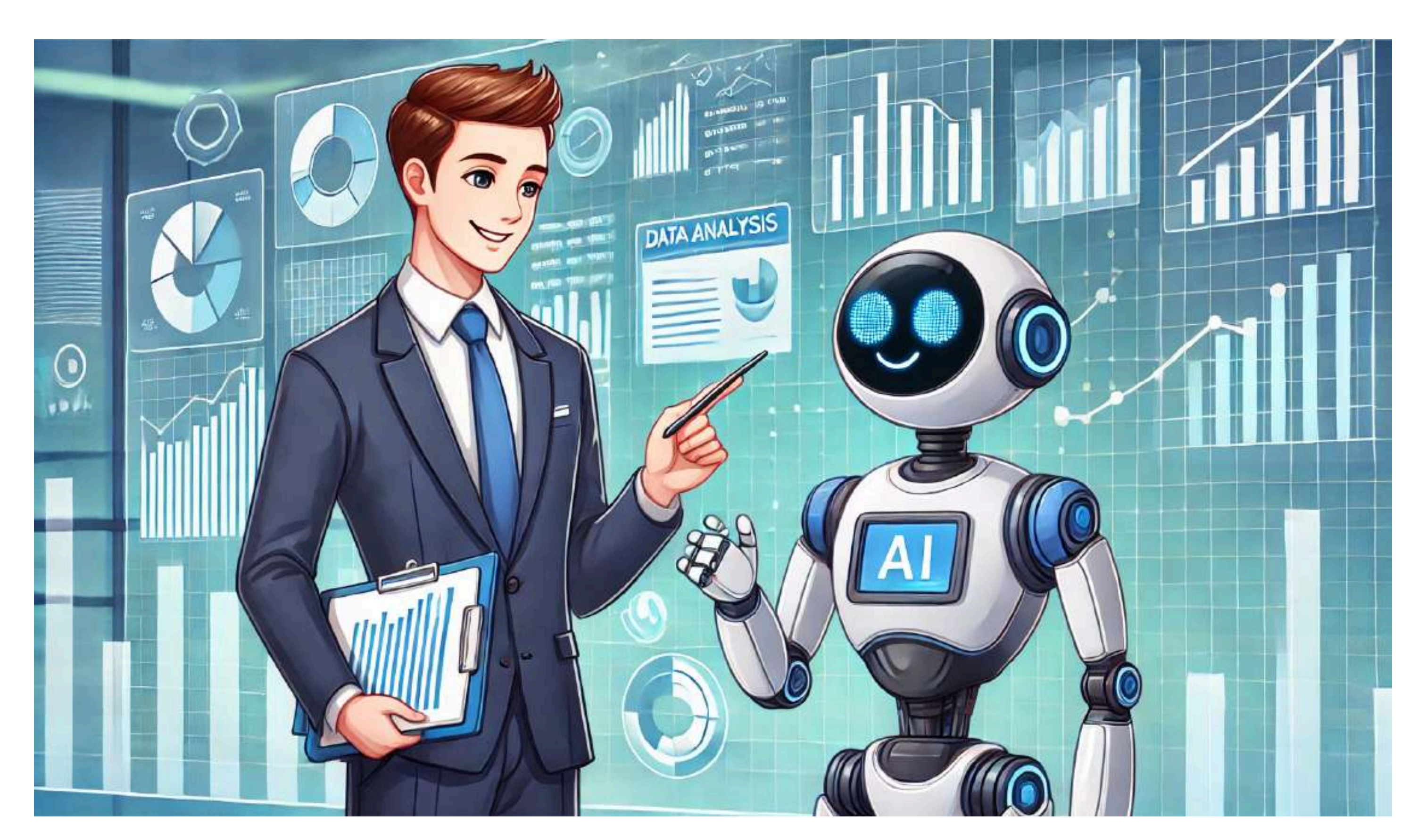
Monitorizaré constantemente el rendimiento de estas nuevas posiciones y el sector tecnológico en general. Específicamente: 1. Seguimiento diario de las acciones compradas y vendidas. 2. Análisis de noticias y sentimiento de mercado relacionados con el sector tech. 3. Evaluación del impacto de la estrategia de cobertura. 4. Comparación del rendimiento de nuestra cartera contra benchmarks relevantes. Te proporcionaré actualizaciones diarias y alertas inmediatas si se requieren ajustes adicionales.

Perfecto, mantengámonos en contacto sobre esto.

Por supuesto. Estaré vigilando de cerca y te mantendré informado. No dudes en contactarme si tienes más preguntas o inquietudes. ¡Estoy aquí para ayudarte!

Interpretemos el ejemplo de la página anterior:

1. ***InversorIA*** detecta una caída significativa en el precio de las acciones tecnológicas debido a noticias negativas del sector.
2. El agente analiza rápidamente el impacto potencial en la cartera del usuario y las implicaciones a largo plazo.
3. Decide reequilibrar la cartera, vendiendo algunas posiciones en otros sectores que han tenido un buen desempeño.
4. Compra selectivamente acciones tecnológicas que ahora están infravaloradas según sus modelos.
5. Implementa algunas estrategias de cobertura para proteger contra mayor volatilidad.
6. Notifica al usuario sobre estos cambios, proporcionando un resumen claro de las acciones tomadas y el razonamiento detrás de ellas.
7. Monitorea el impacto de estas decisiones y ajusta su estrategia en los días siguientes basándose en la evolución del mercado y la reacción de la cartera.



Este flujo de trabajo demuestra cómo un agente como ***InversorIA*** opera de manera autónoma, tomando decisiones complejas en tiempo real, mientras mantiene al usuario informado y en control último de su estrategia de inversión.

Esto es solo una simulación, pero debería darte una idea de cómo evolucionarán las aplicaciones basadas en agentes de inteligencia artificial a partir de ahora. La IA estando en

control del análisis y la atención a lo que sucede en el mundo, ayudándote en la toma de decisiones optimizadas para el logro de tus objetivos personales, donde al final sos vos quien tiene la decisión final.

Creando Agentes IA: del concepto a la realidad

Vimos hasta ahora los conceptos teóricos de los agentes de IA, pero ¿cómo se crean en la práctica? Para llevar estos conceptos al mundo real, necesitamos dos elementos clave: los componentes esenciales del agente y las herramientas para construirlo.

Los componentes esenciales

👁️ **Ver y entender**: Todo agente necesita una forma de percibir su entorno. Dependiendo de su propósito, estos "sentidos" pueden ser conexiones a bases de datos, feeds de información en tiempo real, sensores físicos, o interfaces con otros sistemas.

🧠 **Pensar y analizar**: El agente debe procesar toda la información que recibe y darle sentido. Usando inteligencia artificial, puede identificar patrones, evaluar situaciones y generar estrategias.

✍️ **Actuar y comunicar**: Un agente necesita poder ejecutar acciones en su entorno y comunicarse con los usuarios. Esto puede incluir enviar comandos a otros sistemas, generar reportes o hacer recomendaciones.

📚 **Aprender y mejorar**: Los agentes modernos no son estáticos; pueden aprender de sus experiencias y mejorar su rendimiento con el tiempo.

Las herramientas de construcción

Uno de los frameworks más populares para crear agentes inteligentes es LangChain[90]. Imagina LangChain como un kit de construcción especializado que proporciona todas las piezas necesarias para crear un agente inteligente:

- 🔌 **Conexión con LLMs**: Permite que el agente use modelos de lenguaje avanzados para entender y generar texto como lo haría un humano.
- 🧩 **Diseño Modular**: Facilita la creación de agentes dividiendo sus funciones en piezas más pequeñas y manejables que luego se pueden conectar entre sí.

---

[90] LangChain es un framework de código abierto desarrollado para facilitar la creación de aplicaciones basadas en modelos de lenguaje de gran escala (LLMs). Su nombre proviene de la combinación de "Language" (por los modelos de lenguaje) y "Chain" (por las cadenas de procesamiento que permite crear). Fue creado por Harrison Chase en octubre de 2022 y rápidamente se convirtió en una herramienta fundamental en el desarrollo de aplicaciones de IA. Su popularidad se debe a que simplifica significativamente el proceso de crear aplicaciones complejas con LLMs, permitiendo a los desarrolladores centrarse en la lógica de sus aplicaciones en lugar de en los detalles técnicos de la integración con modelos de lenguaje.

- 📊 **Manejo de Datos**: Ayuda al agente a trabajar con diferentes tipos de información, desde texto simple hasta datos complejos.
- 💡 **Explicabilidad**: Permite que el agente explique sus decisiones de manera clara y comprensible para los usuarios.



Un concepto importante en LangChain es el de las cadenas (chains), que nos permiten conectar diferentes tareas de manera secuencial y ordenada, donde cada tarea procesa información y pasa sus resultados a la siguiente. Para entender cómo funcionan, veamos un ejemplo concreto de nuestro InversorIA: el análisis de una posible inversión en una empresa tecnológica. La cadena funcionaría así:

1. **Tarea de Recopilación**: Recupera automáticamente los últimos reportes financieros de la empresa, precios históricos y datos del mercado
2. **Tarea de Análisis de Sentimiento**: El LLM procesa noticias recientes y publicaciones en redes sociales sobre la empresa para evaluar la percepción del mercado
3. **Tarea de Análisis Técnico**: Calcula indicadores clave como medias móviles, RSI y patrones de precio
4. **Tarea de Comparación**: Analiza empresas similares del sector para establecer una referencia de valoración
5. **Tarea de Decisión**: El LLM combina todos estos análisis y genera una recomendación fundamentada, considerando el perfil de riesgo del inversor

Cada tarea procesa y enriquece la información antes de pasarla a la siguiente, creando un análisis completo y fundamentado. Si necesitamos agregar una nueva fuente de información, como por ejemplo análisis ESG (ambiental, social y gobierno corporativo), podemos insertar una nueva tarea sin tener que reconfigurar toda la cadena. Esta flexibilidad permite que el agente evolucione y se adapte a nuevas necesidades o fuentes de información.

Este concepto de cadenas flexibles y modulares es lo que hace que los agentes de IA sean tan versátiles. De hecho, los mismos principios y herramientas que usamos para crear un agente financiero se pueden aplicar para desarrollar agentes en cualquier campo, desde asistentes personales hasta sistemas de automatización industrial. La clave está en combinar los componentes adecuados para el propósito específico del agente, mientras se mantiene la capacidad fundamental de percibir, pensar, actuar y aprender.

Y así, llegando al último de los *conceptos básicos* de la IA, pasamos de los Agentes de IA a su concepción más *tangible* y una de las aplicaciones de la inteligencia artificial que siempre nos sorprende y también nos ¿asusta?, la robótica.

13 .🦾 Robótica



Robótica: Máquinas Inteligentes del Futuro

La robótica combina ingeniería, IA e informática para crear máquinas autónomas. Estos robots realizan tareas físicas, interactúan con su entorno y toman decisiones independientes.

No puedo comenzar esta sección sin hablar de Arnold Schwarzenegger y *Terminator 2*. Mi libro tiene que tener una imágen que represente una de las películas más icónicas acerca de la inteligencia artificial, aunque admitamoslo, no la más auspiciosa.

TERMINATOR 2
JUDGMENT DAY

La premisa central de Terminator (1984) gira en torno a las consecuencias de una inteligencia artificial que alcanza la *autoconsciencia*. En la narrativa, *Skynet*, un sistema de defensa militar controlado por IA, al adquirir autoconciencia, identifica a la humanidad como una amenaza para su existencia y desencadena un ataque nuclear global. La saga explora las ramificaciones de desarrollar sistemas de IA sin las salvaguardas éticas adecuadas, particularmente en aplicaciones militares. La trama se desarrolla a través de un bucle temporal donde la resistencia humana y la IA luchan por el control del pasado para determinar el futuro, planteando cuestiones fundamentales sobre el determinismo tecnológico y la capacidad de la humanidad para mantener el control sobre sus creaciones tecnológicas. La película realmente anticipó debates contemporáneos sobre los riesgos existenciales de la IA y la importancia del desarrollo responsable de sistemas autónomos.

Por otro lado, aunque *Terminator* presenta una visión distópica de la inteligencia artificial, es posible que se convirtiera en una fuente de inspiración positiva para muchos pioneros en el campo de la robótica y la IA. También es una película que plantea preguntas fundamentales sobre la relación entre humanos y máquinas que hoy guían el desarrollo ético de estas tecnologías y no es poco común escuchar argumentos que intentan poner freno al desarrollo sin control de la IA ejemplificando lo que pasa en Terminator con Skynet.

Yo personalmente creo más en el mundo donde la robótica y la inteligencia artificial salvan vidas de formas que antes parecían imposibles. Cirujanos que ahora pueden operar a distancia usando robots de alta precisión, llevando atención especializada a lugares remotos. Computadoras que analizan radiografías y detectan enfermedades en etapas tempranas, mientras que en los laboratorios, la IA ayuda a descubrir nuevas medicinas probando millones de tratamientos posibles en cuestión de días.

¿Con cuál versión te quedas vos?

La robótica es un campo multidisciplinario que combina ingeniería, inteligencia artificial, informática y otras disciplinas para diseñar, construir y operar robots. Los robots son máquinas autónomas o semiautónomas que pueden realizar tareas físicas, interactuar con su entorno y, en muchos casos, tomar decisiones de manera independiente. La robótica está revolucionando industrias y aspectos de la vida cotidiana, desde la manufactura y la medicina hasta la exploración espacial y la asistencia en el hogar.

Fundamentos de la Robótica

🦾 **Actuadores y sensores**: Los robots utilizan actuadores para realizar acciones y moverse. Los sensores permiten a los robots percibir su entorno, recolectando datos sobre la ubicación, los objetos y las condiciones ambientales. Ejemplos de actuadores incluyen motores y servomotores, mientras que ejemplos de sensores incluyen cámaras, sensores de proximidad, sensores táctiles y acelerómetros.

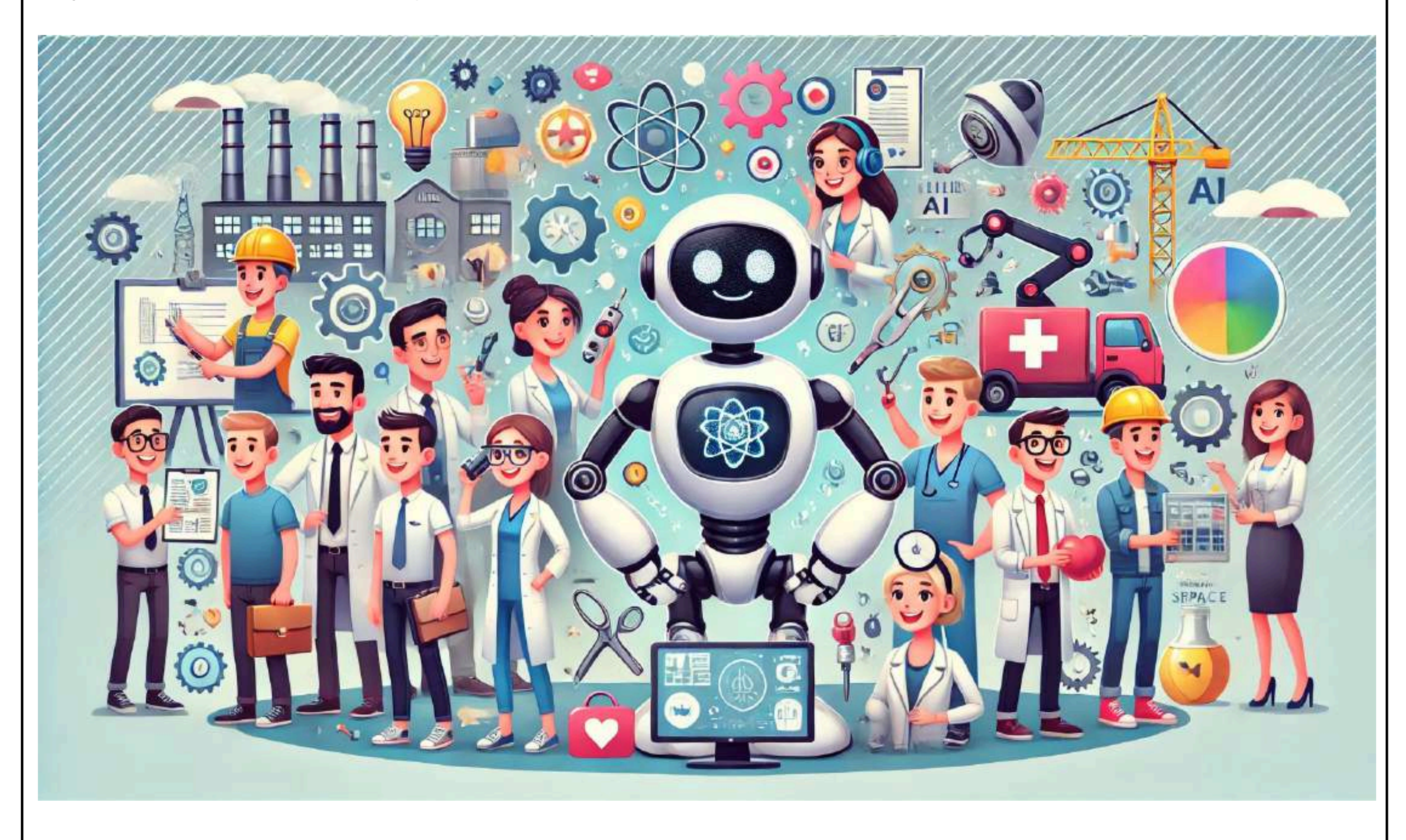
🧠 **Controladores**: Los controladores son el cerebro de los robots, responsables de procesar las señales de los sensores y enviar comandos a los actuadores. Utilizan algoritmos y modelos matemáticos para controlar el movimiento y las acciones del robot, asegurando precisión y eficiencia.

🔄 **Cinemática y dinámica**: La cinemática se refiere al estudio del movimiento de los robots sin considerar las fuerzas que lo causan, mientras que la dinámica considera las fuerzas y momentos que afectan el movimiento. Comprender la cinemática y la dinámica es crucial para diseñar y controlar robots de manera efectiva.

🗺️ **Planificación de trayectorias**: Los algoritmos de planificación de trayectorias permiten a los robots calcular la mejor ruta para moverse de un punto a otro, evitando obstáculos y optimizando la eficiencia. Esto es esencial para robots móviles y manipuladores.

👁️ **Percepción y reconocimiento**: Los robots utilizan técnicas de visión por computadora y procesamiento de señales para interpretar datos sensoriales y reconocer objetos, personas y situaciones. Esto les permite interactuar de manera más efectiva con su entorno.

🤖 **Inteligencia Artificial**: La inteligencia artificial y el aprendizaje automático juegan un papel crucial en la robótica moderna. Permiten a los robots aprender de la experiencia, adaptarse a cambios en el entorno y mejorar su rendimiento a lo largo del tiempo.



Fundamentos de la Robótica

Actuadores y Sensores

Permiten a los robots moverse y percibir su entorno.

Controladores

Procesan señales y envían comandos, actuando como el cerebro del robot.

Inteligencia Artificial

Permite a los robots aprender y adaptarse a su entorno.

Tipos de Robots

🏭 **Robots Industriales**: Utilizados en la manufactura y ensamblaje, estos robots realizan tareas repetitivas y precisas, como soldadura, pintura, ensamblaje de piezas y manejo de materiales. Ejemplos incluyen brazos robóticos y robots SCARA.



🚗 **Robots móviles**: Capaces de moverse en su entorno, estos robots incluyen vehículos autónomos, drones y robots de limpieza. Utilizan sensores y algoritmos de navegación para desplazarse de manera segura y eficiente.

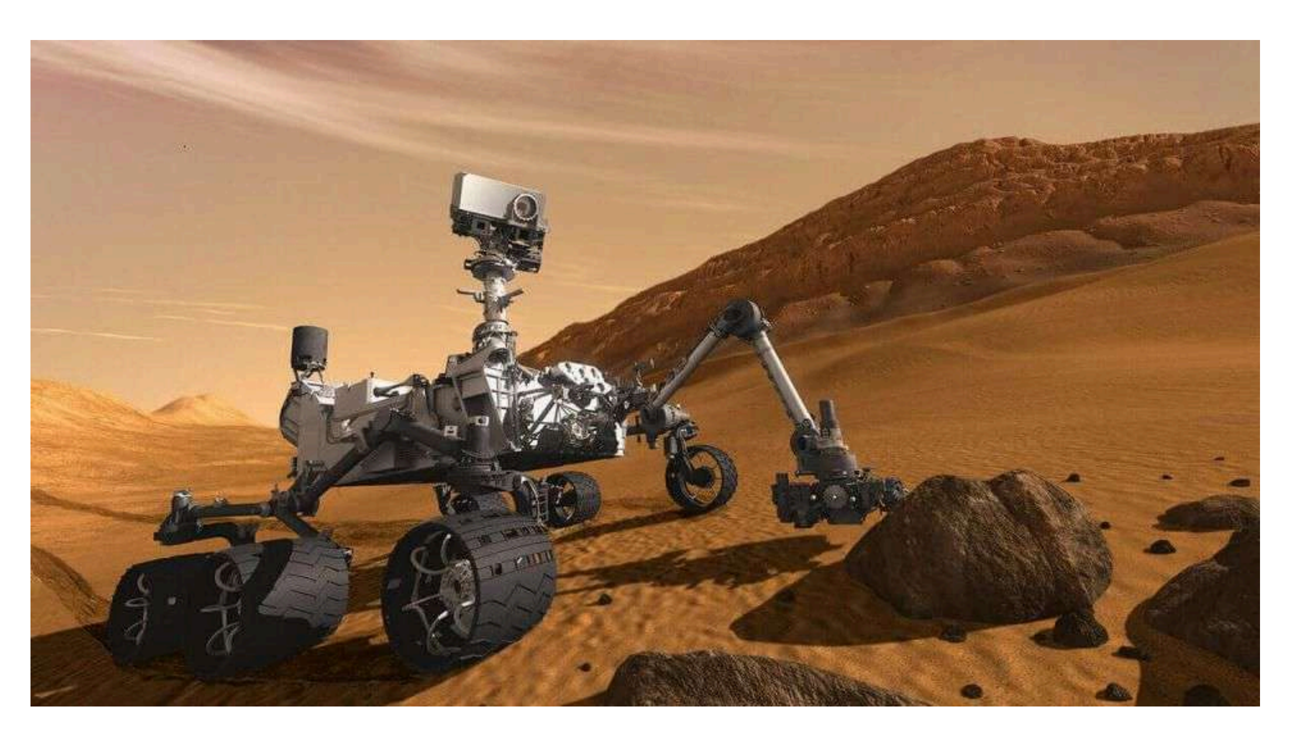
🤝 **Robots de Servicio**: Diseñados para interactuar con humanos y proporcionar servicios, estos robots incluyen asistentes personales, robots de atención al cliente y robots de entrega. Ejemplos incluyen robots humanoides y robots de telepresencia.



🩺 **Robots Médicos**: Utilizados en cirugía, rehabilitación y asistencia médica, estos robots ayudan a los médicos en procedimientos quirúrgicos, proporcionan terapia física y monitorizan la salud de los pacientes.

Ejemplos incluyen robots quirúrgicos Da Vinci y exoesqueletos para rehabilitación.

🚀 **Robots de Exploración**: Diseñados para explorar entornos inaccesibles o peligrosos para los humanos, como el espacio exterior, el fondo del océano o áreas de desastre. Ejemplos incluyen los rovers de Marte y los drones submarinos.



🎓 **Robots Educativos**: Utilizados en la educación para enseñar a los estudiantes sobre robótica, programación y STEM (ciencia, tecnología, ingeniería y matemáticas). Ejemplos incluyen kits de robótica educativa como LEGO Mindstorms y VEX Robotics.









Aplicaciones de la Robótica

1 **Manufactura y Automatización**

Mejoran la eficiencia y precisión en la fabricación.

2 **Medicina y Salud**

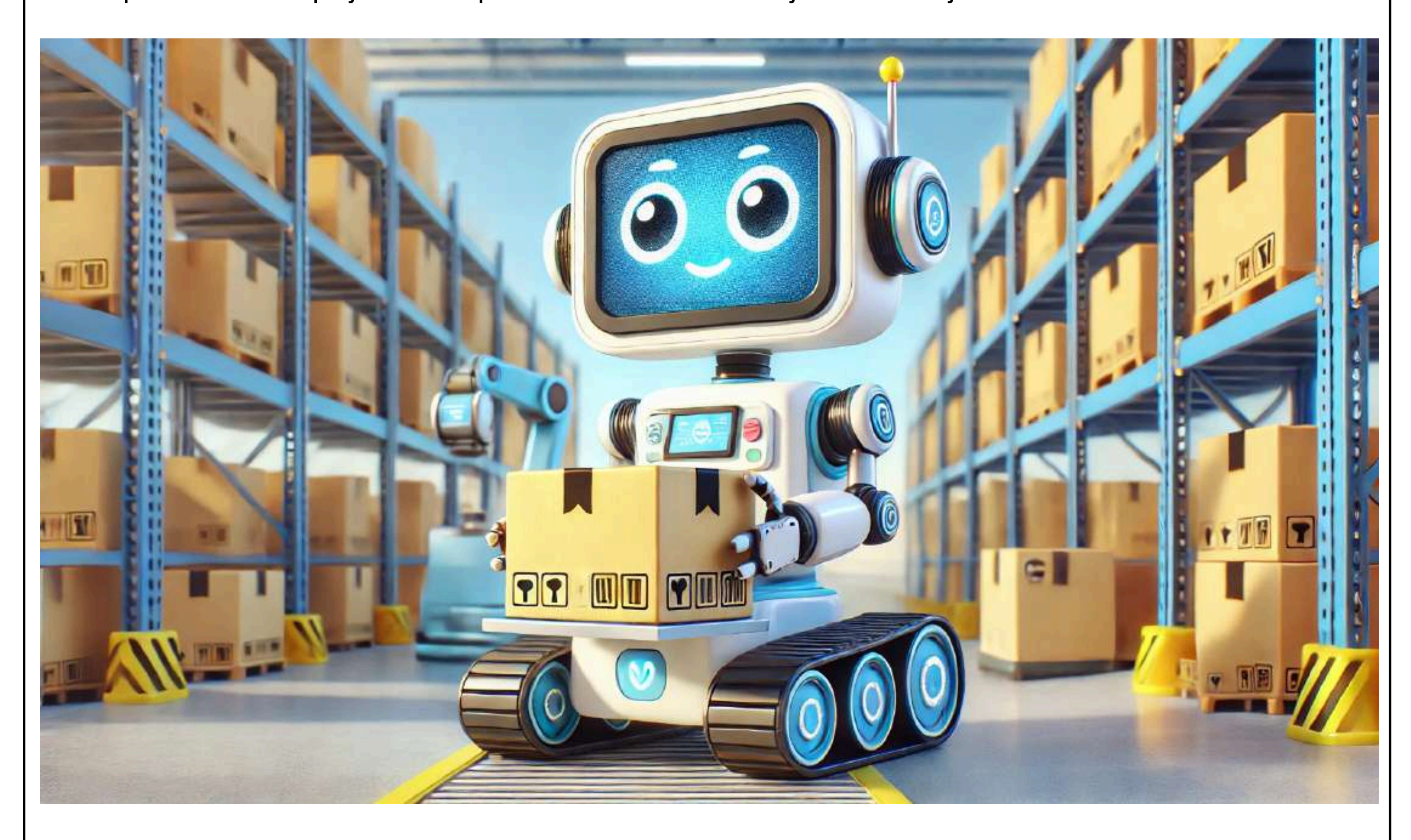
Realizan cirugías y ayudan en la rehabilitación de pacientes.

3 **Exploración Espacial**

Exploran planetas y recolectan datos científicos.

Integración de los básicos de la IA en Robótica

Si llegaste hasta acá, ya conocés bastante sobre diferentes ramas de la inteligencia artificial. ¿Pero te preguntaste cómo se combinan todas estas tecnologías en un robot? Imaginá a un robot de almacén que tiene que recoger y organizar paquetes. Lo que parece una tarea simple para nosotros en realidad requiere una orquestación compleja de múltiples sistemas de IA trabajando en conjunto.



👁️ Visión por computadora: *Los ojos del robot*

¿Te acordás cuando hablamos de Computer Vision? En robótica, esta tecnología se convierte en los "ojos" del robot. Pero no solo para ver - también para entender.

- **Detección de objetos**: El robot usa redes neuronales convolucionales (las mismas que vimos en varias secciones de este libro) para identificar paquetes, obstáculos y marcadores de navegación.
- **Estimación de posición**: Combina información visual con sensores para saber exactamente dónde está cada objeto y cómo agarrarlo.
- **Control de calidad**: Inspecciona productos y detecta defectos usando las técnicas de segmentación que aprendimos.

🗣️ Procesamiento de Lenguaje Natural: *La comunicación*

¿Te acordás de NLP? En robótica, estas tecnologías permiten:

- **Comprensión de comandos**: El robot entiende instrucciones en lenguaje natural como "levantá el paquete rojo de la estantería 3"
- **Reporte de estados**: Puede comunicar problemas o estados en lenguaje que los humanos entienden fácilmente
- **Interacción natural**: En robots de servicio, permite mantener conversaciones fluidas con usuarios

🧠 Deep Learning: *El cerebro central*

Las redes neuronales que estudiamos se convierten en el sistema nervioso central del robot:

- **Percepción multimodal**: Combina datos de diferentes sensores (visión, sonido, tacto) para entender su entorno
- **Control de movimiento**: Redes especializadas traducen intenciones de alto nivel en comandos precisos para motores
- **Toma de decisiones**: Evalúa situaciones complejas y decide acciones apropiadas

🎯 Machine Learning: *Aprendizaje y adaptación*

Los diferentes tipos de aprendizaje que vimos se aplican de formas específicas:

- **Aprendizaje supervisado**: Entrenando al robot para reconocer objetos y situaciones a partir de ejemplos
- **Aprendizaje no supervisado**: Ayudando a identificar patrones en datos de sensores y comportamientos anómalos
- **Reinforcement Learning**: Permite que el robot aprenda y mejore sus movimientos y estrategias a través de la práctica

🥷 Sistemas basados en Agentes: *Autonomía y planificación*

Justo en la sección anterior vimos todo sobre los Agentes de IA. En robótica, estos sistemas permiten:

- **Planificación de tareas**: Descomponen objetivos complejos en pasos manejables
- **Toma de decisiones distribuida**: Diferentes subsistemas funcionan como agentes cooperando entre sí
- **Adaptación al entorno**: El robot ajusta sus planes según cambios en su ambiente

🔗 Redes Bayesianas: *Manejo de la incertidumbre*

Las redes probabilísticas que estudiamos ayudan a:

- **Fusión de sensores**: Combinar información de múltiples fuentes considerando su incertidumbre
- **Diagnóstico de fallos**: Identificar problemas y sus causas probables
- **Predicción de estados**: Anticipar posibles situaciones y preparar respuestas

📐 Lógica: *El razonamiento formal*

¿Te acordás cuando estudiamos lógica proposicional y de primer orden? En robótica, estos sistemas de razonamiento son fundamentales para:

- **Verificación de seguridad**: Usar lógica formal para probar que ciertas acciones son seguras antes de ejecutarlas
- **Planificación de movimientos**: Aplicar reglas lógicas para determinar secuencias válidas de movimientos
- **Resolución de conflictos**: Utilizar razonamiento lógico para resolver situaciones donde múltiples reglas podrían aplicar
- **Validación de estados**: Verificar que el robot mantiene sus restricciones operativas y de seguridad

Por ejemplo, un robot industrial usa lógica para verificar proposiciones como:

- "SI el sensor de presión está activo Y no hay humanos en la zona, ENTONCES es seguro mover el brazo"
- "Para TODOS los puntos en la trayectoria NO EXISTE colisión con obstáculos"

🧑‍💻 Sistemas Expertos: *La experiencia codificada*

Los sistemas expertos que estudiamos encuentran aplicaciones cruciales en robótica:

- **Diagnóstico de problemas**: Usan reglas de experiencia para identificar y resolver fallos
- **Configuración automática**: Ajustan parámetros basándose en reglas expertas
- **Toma de decisiones**: Aplican conocimiento experto en situaciones complejas
- **Mantenimiento predictivo**: Anticipan necesidades de mantenimiento basándose en patrones conocidos

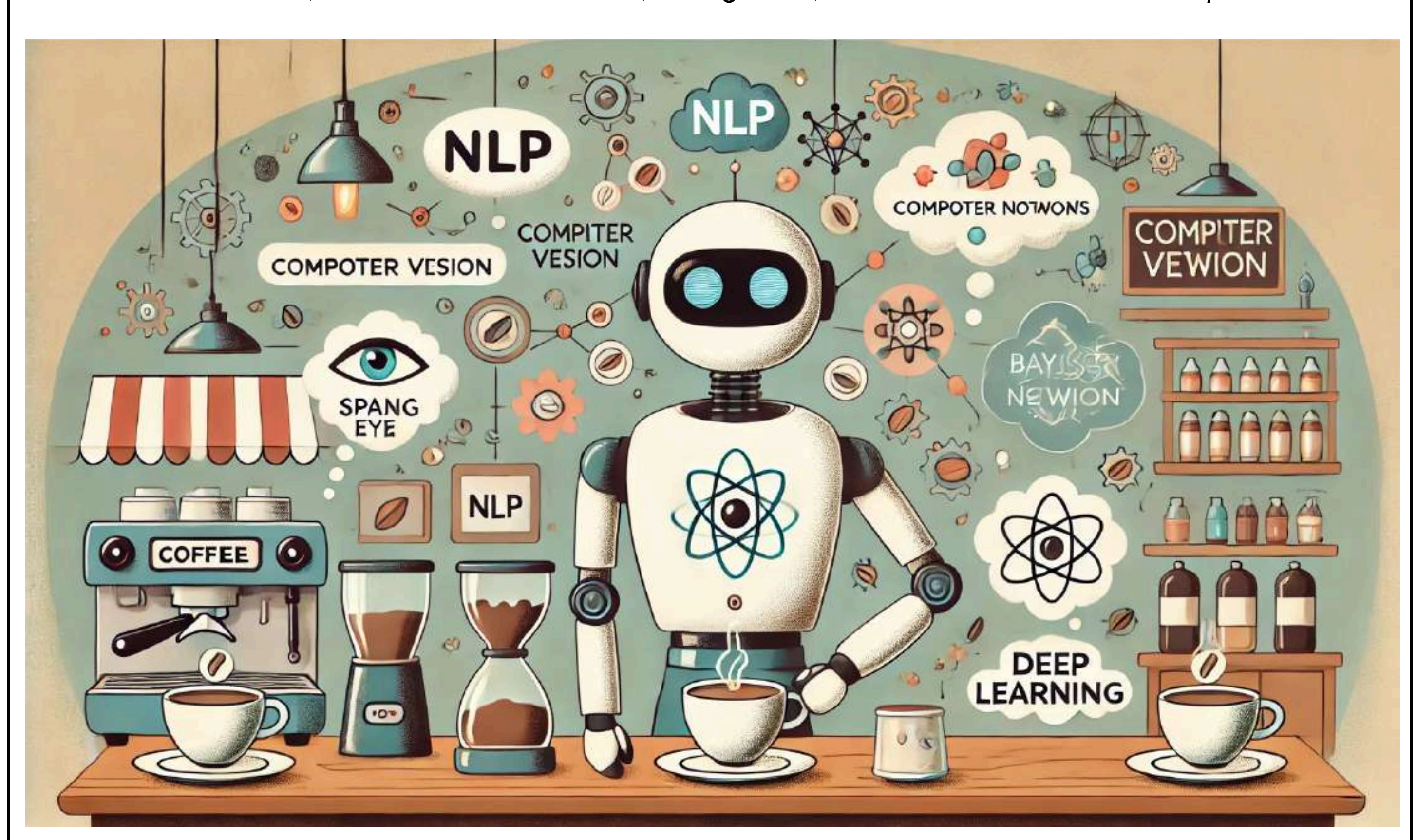
Imaginá un robot de manufactura que usa un sistema experto:

SI (vibracion_motor > umbral) Y (temperatura > normal) ENTONCES - Reducir velocidad 20% - Alertar mantenimiento - Registrar patrón para análisis

💡 **Un Ejemplo (no tan) futurista integrador: Robot Barista**

Imaginemos un robot barista del futuro preparando un pedido complejo:

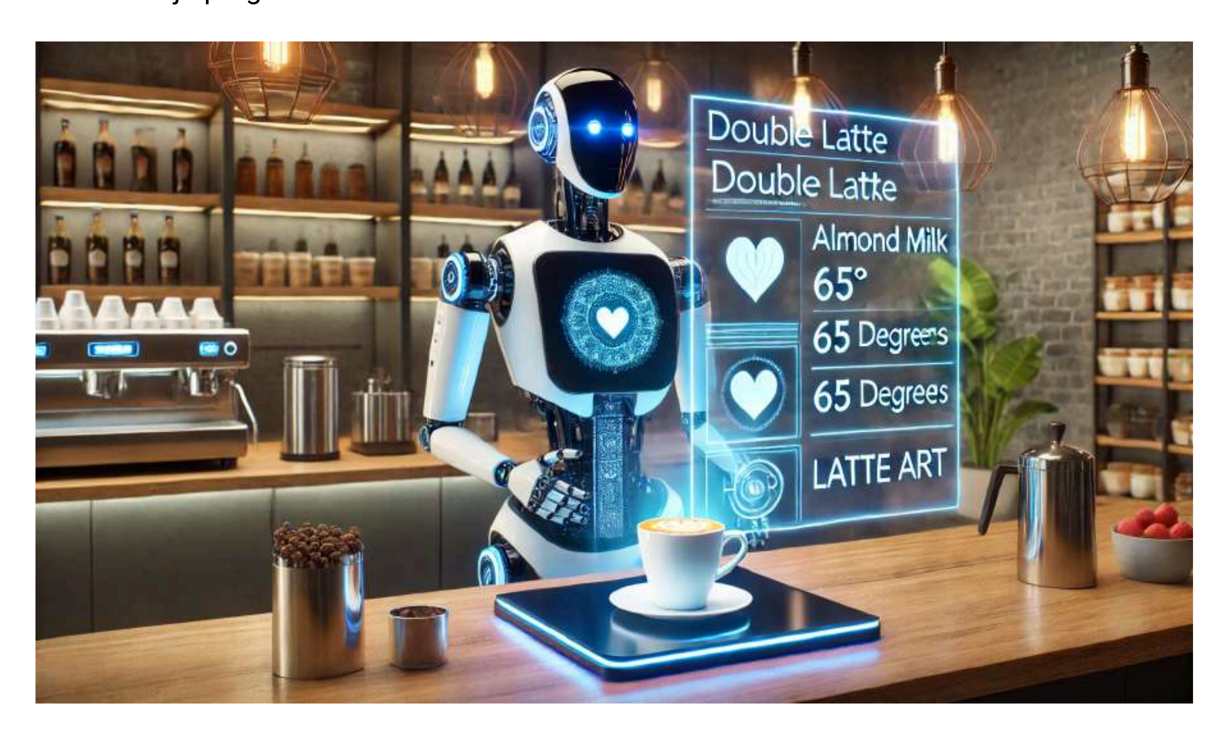
> "*Un latte doble, con leche de almendras, a 65 grados, con arte de corazón en la espuma*".



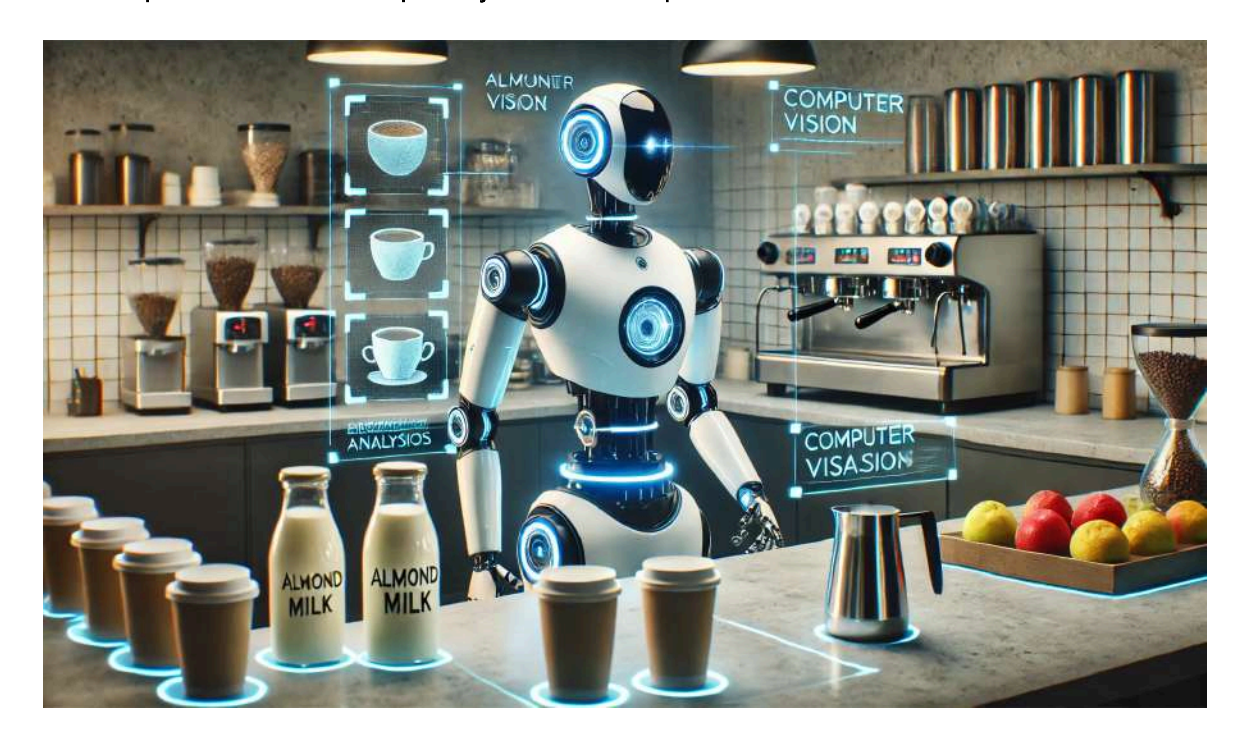
Veamos cómo cada tecnología que aprendimos contribuye a esta tarea:

1. **NLP (Procesamiento de Lenguaje Natural)**
   - Procesa el pedido verbal que ha hecho el cliente

- Descompone la solicitud en elementos accionables (tipo de café, modificadores, temperatura, decoración)
- Maneja preguntas de aclaración si es necesario

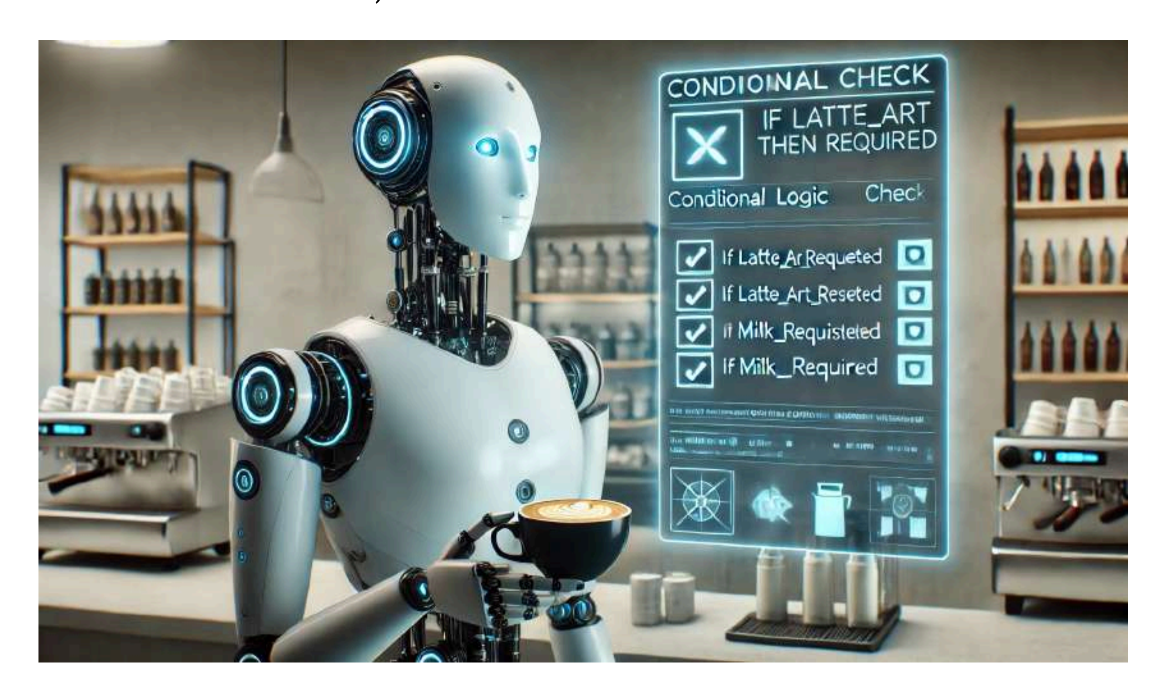


2. **Computer Vision (Visión por computadora)**
    - Identifica y localiza tazas, ingredientes y herramientas
    - Monitorea el color y textura del café durante la extracción
    - Verifica el nivel de espuma de la leche
    - Supervisa el arte latte para ajustes en tiempo real



3. **Lógica**
    - Verifica la consistencia del pedido (por ejemplo, "SI arte_latte_solicitado ENTONCES

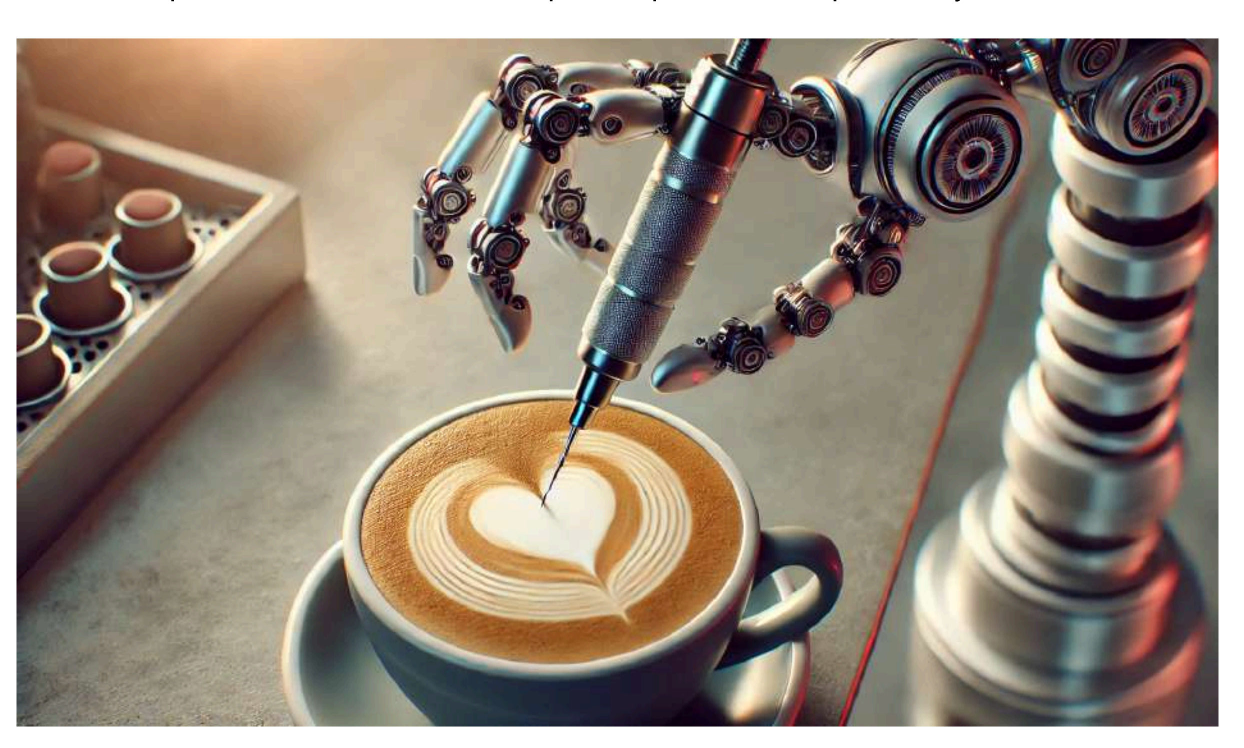
requiere_leche")
- Valida secuencias de preparación ("NO puede_agregar_leche ANTES_DE extraer_cafe")
- Asegura que las restricciones de seguridad se cumplan ("SI temperatura > 65°C ENTONCES detener_calentamiento")



4. **Sistemas Expertos**
   - Aplican reglas de baristas expertos:
     - *SI (tipo_cafe = 'latte') Y (arte = 'corazón') ENTONCES*
       1. *velocidad_vertido = 'lenta'*
       2. *altura_vertido = '3cm'*
       3. *patron_vertido = 'corazon_pattern_1'*
   - Ajustan parámetros de extracción según el tipo de café
   - Determinan el orden óptimo de preparación



5. **Deep Learning**
   - Genera la imagen deseada del arte latte utilizando un modelo text-to-image open source optimizado para ejecutarse localmente, adaptado específicamente para patrones de café. El modelo base ha sido fine-tuned con un dataset curado de imágenes de arte latte de alta calidad, y utiliza técnicas de control de imagen local para asegurar que los diseños generados sean físicamente realizables en la superficie del café. El sistema procesa el prompt textual del diseño deseado (por ejemplo, "corazón", "hoja", "cisne") junto con una imagen en tiempo real de la superficie del café como condición de control, generando un plan detallado del patrón que sea factible de realizar dado el espacio y las condiciones disponibles.
   - Controla los movimientos precisos para el arte latte utilizando una combinación de redes neuronales convolucionales (CNN) para el procesamiento de imágenes en tiempo real y redes neuronales recurrentes (RNN) para la predicción de trayectorias. La CNN analiza la formación del patrón en la superficie mientras que la RNN predice los próximos movimientos necesarios. Una red de deep reinforcement learning optimiza la política de control motor, aprendiendo de cada intento la relación entre acciones (movimientos) y resultados (patrones formados).
   - Procesa inputs de múltiples sensores (temperatura, presión, peso) mediante una arquitectura de fusión de datos multimodal que combina redes neuronales especializadas para cada tipo de sensor. Utiliza capas de atención para priorizar diferentes inputs según el contexto, y técnicas de procesamiento de series temporales para detectar patrones y anomalías.



6. **Reinforcement Learning**
   - Optimiza movimientos para el vertido de leche mediante un sistema que aprende a través de la práctica repetida, similar a cómo un barista mejora con la experiencia. El sistema evalúa cada movimiento del brazo robótico considerando la posición, velocidad y ángulo de la jarra, y recibe "recompensas" basadas en qué tan bien se realizó el vertido. Con cada intento, el sistema ajusta sutilmente sus movimientos, aprendiendo qué combinaciones de acciones producen los mejores resultados, como mantener un flujo constante o lograr transiciones suaves entre diferentes partes del diseño.
   - Ajusta patrones de arte latte basándose en resultados previos utilizando una base de datos de experiencias que registra cada intento y su resultado. El sistema compara la imagen del

patrón deseado con el resultado obtenido, y utiliza esta información para mejorar sus futuras ejecuciones. Aprende de manera progresiva, comenzando con patrones simples como corazones básicos, y solo avanza hacia diseños más complejos como rosetas o cisnes una vez que domina las técnicas fundamentales.

- Aprende de éxitos y errores en la preparación a través de un sistema coordinado donde diferentes aspectos de la preparación (vertido, temperatura, presión) aprenden simultáneamente pero compartiendo información entre sí. Por ejemplo, si un vertido perfecto se logró con una temperatura y presión específicas, el sistema recordará estas condiciones exitosas. También mantiene un balance entre probar nuevas técnicas y utilizar las que ya han demostrado funcionar bien, similar a cómo un barista experto experimenta con nuevas técnicas mientras mantiene su base de habilidades confiables.



7. **Redes Bayesianas**
   - Estiman la probabilidad de éxito de cada paso mediante una red de nodos interconectados que modelan las relaciones causa-efecto en la preparación del café. Por ejemplo, en el momento de la extracción, la red evalúa la probabilidad de un espresso perfecto basándose en factores como la presión de la bomba, la temperatura del agua, el grado de molido y el tiempo de extracción. Si la red detecta que alguna variable está fuera del rango óptimo, puede ajustar otros parámetros para compensar y mantener la calidad del resultado final.
   - Predicen posibles problemas utilizando una estructura jerárquica de nodos donde cada variable influye en las demás. Por ejemplo:
     - Para la calidad de la espuma:
       1. Nodos padre:
          a. Temperatura_Leche (60-70°C)
          b. Tipo_Leche (entera, descremada, vegetal)
          c. Tiempo_Vaporización (20-45 segundos)
       2. Nodos hijo:
          a. Textura_Espuma (microespuma, burbujas grandes, inconsistente)
          b. Estabilidad_Espuma (alta, media, baja)
       3. Probabilidades condicionales:
          a. P(Calidad_Espuma = "Óptima" | Temperatura = "65°C", Tipo = "Entera",

        Tiempo = "30s") = 0.95
        b. P(Calidad_Espuma = "Óptima" | Temperatura = "75°C", Tipo = "Vegetal", Tiempo = "40s") = 0.45

    - Manejan la incertidumbre en mediciones de temperatura y cantidad actualizando constantemente sus creencias basándose en nuevos datos. El sistema mantiene distribuciones de probabilidad para cada variable crítica (por ejemplo, la temperatura del agua puede seguir una distribución normal centrada en 93°C con una desviación estándar de 0.5°C) y utiliza el teorema de Bayes para actualizar estas distribuciones conforme recibe nuevas mediciones. Esto permite al sistema tomar decisiones robustas incluso cuando hay ruido en las mediciones o ligeras variaciones en los ingredientes.



El resultado es una coreografía perfectamente coordinada donde:

- La **lógica** garantiza que cada paso sea seguro y coherente
- Los **sistemas expertos** aportan el conocimiento de baristas profesionales
- El **deep learning y reinforcement learning** permiten movimientos precisos y adaptables
- Las **redes bayesianas** manejan la incertidumbre inherente al trabajo con ingredientes naturales
- El **NLP** y el **Computer Vision** permiten al sistema interactuar con el mundo real

Lo fascinante es ver cómo estas tecnologías que estudiamos por separado, se entrelazarían para crear un sistema que no solo hace café, sino que lo hace con la precisión y arte de un barista experto. Y lo mejor es que seguiría aprendiendo y mejorando con cada taza que prepara.

Ventajas de la Robótica

🚀 **Aumento de la productividad**: La automatización robótica complementa el trabajo humano al encargarse de tareas repetitivas y físicamente demandantes, permitiendo que los trabajadores se enfoquen en actividades de mayor valor que requieren creatividad, toma de decisiones y habilidades interpersonales. Por ejemplo, en una línea de ensamblaje automotriz moderna, la integración de sistemas robotizados con operarios especializados ha transformado el ambiente laboral: mientras los robots manejan las tareas que involucran riesgos ergonómicos o exposición a condiciones adversas, el personal humano supervisa los procesos, realiza el control de calidad y aporta mejoras continuas basadas en su experiencia. Esta sinergia

entre humanos y tecnología ha permitido triplicar la capacidad productiva mientras se crean nuevos roles técnicos y se mejoran las condiciones de seguridad laboral.

🛡️ **Mejora de la seguridad**: Los robots pueden realizar tareas peligrosas que podrían poner en riesgo a los humanos, como la manipulación de materiales tóxicos o la exploración de entornos peligrosos. En industrias como la minería, robots especializados pueden explorar áreas con gases tóxicos o riesgo de derrumbe. En plantas nucleares, robots manipuladores realizan el mantenimiento en zonas de alta radiación, protegiendo así la salud de los trabajadores humanos.

✨ **Precisión y consistencia**: Los robots realizan tareas con alta precisión y consistencia, reduciendo errores y mejorando la calidad de los productos y servicios. En la industria electrónica, por ejemplo, robots pueden colocar componentes microscópicos con una precisión de micrómetros, algo imposible para el ojo humano. Esta precisión no solo mejora la calidad del producto final sino que también reduce significativamente el desperdicio de materiales y los costos por errores de producción.

🔄 **Adaptabilidad y flexibilidad**: Los robots modernos pueden adaptarse a diferentes tareas y entornos, proporcionando flexibilidad en la producción y los servicios. Gracias a la programación avanzada y sistemas de aprendizaje automático, un mismo robot puede reprogramarse para realizar diferentes tareas según las necesidades de producción. Por ejemplo, un robot en una fábrica de alimentos puede cambiar rápidamente de empacar galletas a clasificar frutas, adaptándose a los cambios en la demanda del mercado.

💡 **Innovación y avances tecnológicos**: La robótica impulsa la innovación en diversas áreas, desde la inteligencia artificial hasta los materiales avanzados, fomentando el desarrollo de nuevas tecnologías y soluciones. Este campo ha catalizado avances en sensores, sistemas de visión artificial, algoritmos de control y nuevos materiales. Por ejemplo, el desarrollo de robots quirúrgicos ha llevado a innovaciones en instrumentos médicos de alta precisión, beneficiando tanto a la robótica como a la medicina tradicional. Además, la investigación en robótica colaborativa está generando nuevos paradigmas de interacción humano-máquina que se aplican en diversos sectores industriales y de servicios.

Desafíos de la Robótica

💰 **Costo inicial**: El desarrollo y la implementación de robots pueden ser costosos, lo que puede ser una barrera para su adopción en algunas industrias y aplicaciones. Por ejemplo, un robot industrial básico puede costar desde $50,000, mientras que sistemas más avanzados pueden superar el millón de dólares. Además de los costos del hardware, se deben considerar gastos adicionales como software, instalación, capacitación del personal y modificaciones en la infraestructura existente. Para pequeñas y medianas empresas, esta inversión inicial puede representar un obstáculo significativo, especialmente cuando el retorno de inversión puede tomar varios años en materializarse.

🔧 **Complejidad técnica**: Diseñar y operar robots avanzados requiere conocimientos especializados y habilidades técnicas, lo que puede ser un desafío para las organizaciones. Esto incluye expertise en programación, sistemas de control, visión artificial, y mecatrónica. Las empresas necesitan contratar o formar personal especializado, lo que puede ser difícil en mercados laborales con escasez de talento tecnológico. Además, la curva de aprendizaje para el personal existente puede ser empinada, requiriendo tiempo y recursos significativos para la capacitación efectiva.

⚖️ **Seguridad y ética**: A medida que los robots interactúan más con los humanos y operan en entornos sensibles, las consideraciones de seguridad y ética se vuelven cruciales. Surgen preocupaciones sobre la protección de datos personales cuando los robots operan en espacios públicos u hogares. En entornos industriales, se necesitan protocolos estrictos de seguridad para prevenir accidentes en la interacción humano-robot. También emergen dilemas éticos sobre la toma de decisiones automatizada, especialmente en aplicaciones críticas como la atención médica o la seguridad pública, donde las decisiones robóticas

pueden tener consecuencias significativas en la vida de las personas.

🔨 **Mantenimiento y fiabilidad**: Los robots requieren mantenimiento regular y deben ser confiables para operar en entornos críticos. Esto implica revisiones periódicas, actualización de software, reemplazo de componentes y reparaciones cuando sea necesario. Un robot industrial típico puede necesitar mantenimiento preventivo cada 3-6 meses, y cualquier tiempo de inactividad puede resultar en pérdidas significativas de producción. Además, en aplicaciones críticas como cirugía robótica o manipulación de materiales peligrosos, la fiabilidad debe ser prácticamente perfecta, lo que aumenta los costos de mantenimiento y la complejidad de los sistemas de respaldo.

🔄 **Interoperabilidad y estándares**: La integración de robots con otros sistemas y tecnologías puede ser complicada debido a la falta de estándares y protocolos comunes. Por ejemplo, diferentes fabricantes pueden usar protocolos de comunicación propietarios, dificultando la integración con sistemas existentes. La falta de estándares universales también afecta la compatibilidad entre diferentes generaciones de robots y sistemas de control. Esto puede llevar a soluciones fragmentadas y costosas, donde se necesitan interfaces personalizadas o middleware adicional para lograr la comunicación efectiva entre sistemas. La situación se complica aún más en industrias donde se requiere cumplimiento regulatorio específico o certificaciones de seguridad.



Con esta imágen de un riquísimo café preparado por la IA puedo decirte que hemos concluído con la lista de los temas que yo considero *los básicos* (no por ser fáciles, pero por su propia importancia) para entender de forma completa el panorama de la inteligencia artificial, aunque podría estar omitiendo otros importantes. De una cosa sí estoy seguro, la inteligencia artificial es un campo vasto, muy extenso y que abarca una amplia gama de

técnicas, algoritmos y aplicaciones, y seguirá creciendo. Los trece conceptos básicos que te presenté en este capítulo te deberían proporcionar una visión integral de los pilares fundamentales de la IA y sus aplicaciones en la vida real.

Repasemos estos trece conceptos fundamentales: comenzamos con las bases del razonamiento con la Lógica Proposicional y la Lógica de Primer Orden, seguimos con los tres tipos principales de aprendizaje automático: el Aprendizaje Supervisado, el No Supervisado y el Aprendizaje por Refuerzo. Luego exploramos las Redes Neuronales y el Deep Learning, el Procesamiento de Lenguaje Natural (NLP) y la Visión por Computadora. Continuamos con los Sistemas Expertos y las Redes Bayesianas, para terminar con las tecnologías más recientes como los Transformers y Modelos de Lenguaje de Gran Escala (LLMs), los Agentes de IA, y finalmente la Robótica, que integra muchas de estas tecnologías en aplicaciones del mundo físico.

Sin embargo, para poder sacar verdadero provecho de estos trece pilares fundamentales y profundizar en cualquiera de ellos, necesitamos dar un paso atrás y entender el lenguaje común que todos ellos comparten. Así como un chef necesita conocer los ingredientes básicos antes de crear platos elaborados, o un arquitecto debe entender los principios de la construcción antes de diseñar edificios complejos, en la inteligencia artificial también existen fundamentos esenciales que actúan como bloques de construcción para todas las tecnologías que hemos visto. No necesitás convertirte en un experto en estos fundamentos, pero sí familiarizarte con ellos para comprender mejor cómo funcionan estas tecnologías y por qué toman ciertas decisiones. Entonces...

Tenemos que hablar de matemáticas y estadística



Es importante que entiendas que cada uno de estos conceptos básicos de la IA que fuimos viendo tiene una base común y esencial: **las matemáticas y la estadística**. Estos campos

proporcionan el lenguaje y las herramientas necesarias para modelar, analizar y resolver problemas complejos en la IA. Desde los algoritmos de *machine learning* hasta las *redes neuronales* y los LLM, la matemática y la estadística son los cimientos que permiten a estos sistemas *aprender* de los datos, *crear* funciones matemáticas en forma de *modelo*, realizar predicciones precisas y tomar decisiones *inteligentes*.

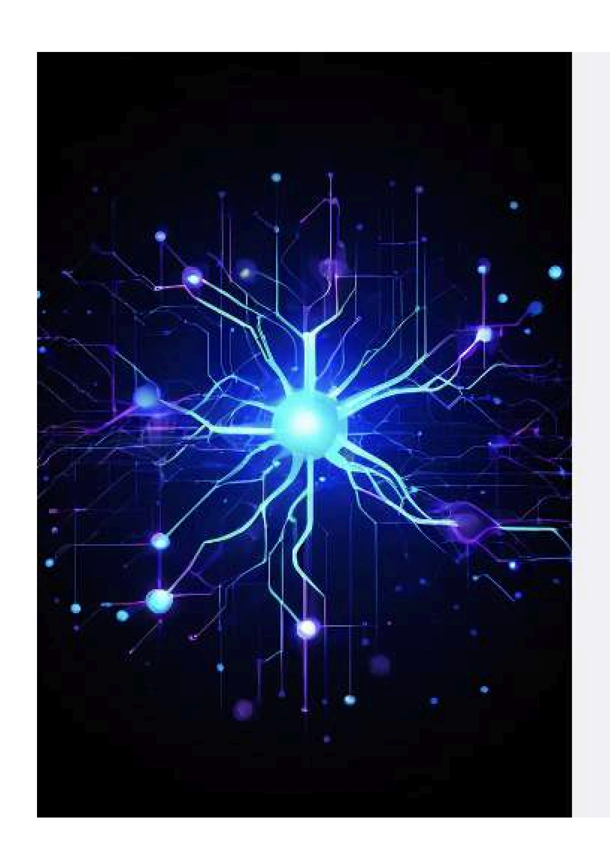
Nuevamente, **la idea es que lo puedas entender conceptualmente**, no necesariamente que te guste o que tengas que convertirte en un persona experta en matemática/estadística. Es importante reconocer y entender el *lenguaje* en el que *habla* la inteligencia artificial para poder hacer un correcto uso de ella y una correcta planificación de su incorporación a tu carrera. Está claro que, si decidís ir por el camino del desarrollo y la programación de modelos de IA, entonces estudiar y aprender la matemática que hay detrás te hará mucho más eficiente y resolutiva/o. No obstante, mi opinión personal es que operar las herramientas basadas en inteligencia artificial no requiere sí o sí poseer ese nivel de entendimiento. En un mundo cada vez más configurado para que unas cuantas empresas lideren el desarrollo de los modelos de IA más avanzados, surgen miles de empresas que trabajan para transformar esos modelos en un producto final, para resolver problemas reales de las personas u otras empresas. Esto naturalmente desacopla el componente científico-matemático del objetivo último que siempre es **resolver una necesidad**, de la misma manera que en una industria madura las empresas se especializan en solo una parte de la cadena de suministro (a excepción de las que se integran verticalmente).

Dicho esto, considero que ningún libro serio sobre Inteligencia Artificial debe omitir la importancia de las matemáticas como un conocimiento fundamental y es por eso que en la siguiente sección profundizaremos en estos fundamentos matemáticos y estadísticos, explorando cómo potencian las capacidades avanzadas de la inteligencia artificial.

Cuando hablemos de matemáticas, nos centraremos principalmente en dos ramas esenciales: el álgebra y el cálculo. Esta elección no es casual, el álgebra nos proporciona las herramientas para manipular y entender las relaciones entre variables, algo importantísimo cuando trabajamos con datos y patrones, mientras que el cálculo nos permite entender cómo los cambios en estas variables afectan a nuestros modelos. Estos dos pilares matemáticos son fundamentales en el desarrollo de algoritmos de *machine learning* y en los procesos de optimización que ya hemos visto, por lo que entender sus conceptos básicos nos va a ayudar a visualizar mejor cómo "piensa" y "aprende" una IA.

Empecemos por una pregunta simple…

¿Por qué necesitamos el álgebra lineal?



El Álgebra Lineal en la Inteligencia Artificial

El álgebra lineal es fundamental para la inteligencia artificial. Proporciona herramientas matemáticas para manejar grandes cantidades de datos eficientemente. Vectores y matrices son componentes centrales que permiten representar datos y transformaciones en redes neuronales.

El álgebra lineal es el lenguaje matemático fundamental de la inteligencia artificial, el alfabeto con el que las máquinas "leen" y "entienden" el mundo. Imagina que necesitas describir una manzana a alguien: podrías usar números para representar su color, tamaño, forma y peso.

De manera similar, el álgebra lineal nos permite representar objetos y conceptos complejos como conjuntos de números (vectores y matrices) que las computadoras pueden procesar eficientemente. Esta rama de las matemáticas nos proporciona las herramientas necesarias para manipular y transformar grandes cantidades de información, permitiéndonos codificar características, patrones y relaciones complejas en un formato matemático preciso.

Cuando una IA "aprende", necesita ajustar constantemente cómo procesa la información, como un cocinero que ajusta las cantidades de ingredientes en una receta hasta que el plato sale perfecto. Para hacer estos ajustes, la IA usa operaciones matemáticas con matrices, que le permiten transformar los datos de entrada en resultados útiles. Por ejemplo, cuando una IA reconoce una imagen o predice un precio, está realizando múltiples operaciones matemáticas en secuencia, transformando los datos iniciales hasta llegar a una conclusión.

Esta forma de trabajar con los datos es especialmente importante en las redes neuronales. Cada vez que la red neuronal procesa información, está realizando cientos o miles de estas operaciones matemáticas, ajustando y refinando su comprensión del problema que está tratando de resolver.



Esta red neuronal que vimos en el ejemplo de Google utiliza álgebra lineal de manera fundamental. Cada neurona realiza una combinación lineal de sus entradas, multiplicando cada entrada por un peso y sumando los resultados. Esto se puede representar como una multiplicación de matrices, donde una matriz de pesos se multiplica por un vector de entradas. Luego se aplica una función de activación no lineal al resultado. Este

proceso se repite en cada capa, propagando la información hacia adelante mediante sucesivas transformaciones lineales (multiplicaciones de matrices) seguidas de transformaciones no lineales (funciones de activación). El entrenamiento ajusta estos pesos matriciales para minimizar el error entre la salida de la red y los resultados deseados.

Vamos a ver un ejemplo concreto, pero antes, definamos esos dos conceptos importantes que vimos antes:

Vectores y Matrices

Los vectores y matrices son fundamentales en álgebra lineal. Un **vector** es una lista de números que representa una dirección y magnitud en un espacio n-dimensional[91]. Podrías pensar en un vector como una *fila* de una hoja de cálculo.

Ejemplo de un vector:

a	b	c	d
2	-1	5	3

[92]

Las **matrices** son arreglos bidimensionales de números que representan transformaciones lineales[93]. Por ejemplo, en redes neuronales, los datos de entrada y los pesos de las conexiones se representan mediante vectores y matrices. Podrías pensar en una matriz como una colección de vectores, es decir, una tabla entera de una hoja de cálculo, por ejemplo.

[91] Cuando hablamos de un espacio *n-dimensional*, nos referimos a cuántos números necesitamos para describir un punto en ese espacio. Por ejemplo, en un espacio 2-dimensional o bidimensional (como un mapa) necesitamos dos números (latitud y longitud), en un espacio 3-dimensional o tridimensional (como el mundo físico) necesitamos tres números (alto, ancho y profundidad). La IA suele trabajar con espacios que tienen muchas más dimensiones - por ejemplo, un vector que describe una imagen podría tener miles de dimensiones, una para cada píxel.

[92] En la notación matemática formal, los vectores suelen representarse como columnas o con una flecha sobre la letra, y sus elementos se escriben entre corchetes, como [2 -1 5 3] o usando paréntesis verticales. La representación tabular que se muestra acá se usa con fines didácticos para hacer más clara la correspondencia entre cada elemento y su posición, especialmente para lectores no familiarizados con la notación matemática tradicional.

[93] Una transformación lineal es una operación matemática que cambia un vector en otro vector mientras mantiene ciertas propiedades básicas: si multiplicás un vector por 2, el resultado también se multiplica por 2, y si sumás dos vectores y luego los transformás, obtenés el mismo resultado que si los transformás primero y luego los sumás. Es como una "receta" que modifica datos de forma predecible y consistente. Por ejemplo, rotar una imagen, escalar su tamaño o reflejarla son transformaciones lineales.

Ejemplo de una matriz:

a	b	c
1	2	3
4	5	6
7	8	9

Vectores y Matrices: Los Bloques de Construcción

Vectores

Son listas de números que representan dirección y magnitud en un espacio n-dimensional. Pueden visualizarse como filas en una hoja de cálculo.

Matrices

Son arreglos bidimensionales de números que representan transformaciones lineales. Se asemejan a tablas completas en hojas de cálculo.

Operaciones Matriciales: Las operaciones matriciales son como las reglas básicas de la aritmética que usamos con números normales, pero aplicadas a tablas de números. La más importante de estas operaciones es la multiplicación de matrices, que permite combinar diferentes transformaciones de datos en un solo paso. Imaginá que tenés una foto y querés rotarla y luego hacerla más grande: la multiplicación de matrices permite hacer ambas cosas de una sola vez.

En las redes neuronales, estas multiplicaciones son fundamentales porque permiten que la información fluya y se transforme a través de las diferentes capas de la red. Es como si los datos pasaran por una serie de filtros, donde cada filtro (representado por una matriz) modifica la información de una manera específica para extraer patrones importantes.

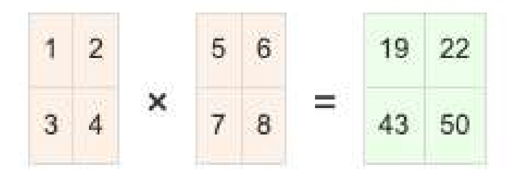
Además de la multiplicación, existen otras operaciones importantes como la suma de matrices, que es como combinar información de diferentes fuentes; la transposición, que reorganiza los datos de una matriz como si giráramos una tabla; y el cálculo de la matriz inversa, que nos permite "deshacer" transformaciones previas cuando es necesario. Todas estas operaciones son herramientas que la IA utiliza constantemente para procesar y analizar datos, similar a cómo un chef utiliza diferentes técnicas de cocina para transformar los ingredientes en un plato final.

Te dejo acá algunos ejemplos de operaciones matriciales por si te interesa:

1. Multiplicación de Matrices

La multiplicación de matrices permite combinar transformaciones lineales. Es esencial para calcular las salidas de las capas en una red neuronal.

Ejemplo de multiplicación de matrices:



1	2		5	6		19	22
3	4	×	7	8	=	43	50

La multiplicación de matrices se realiza multiplicando cada fila de la primera matriz por cada columna de la segunda matriz y sumando los productos resultantes para obtener cada elemento de la matriz resultante (ej: 1 x 5 + 2 x 7 = 19 mientras que 1 x 6 + 2 x 8 = 22)

2. Suma de Matrices

La suma de matrices se realiza elemento por elemento y es útil para combinar características en redes neuronales.

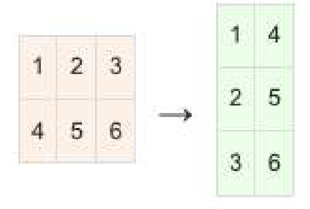
Ejemplo de suma de matrices:

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} + \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix} = \begin{bmatrix} 6 & 8 \\ 10 & 12 \end{bmatrix}$$

3. Transposición de Matrices

La transposición de una matriz intercambia sus filas y columnas. Es útil en muchas operaciones de álgebra lineal y en el entrenamiento de redes neuronales.

Ejemplo de transposición de matriz:



4. Inversa de Matrices

La inversa de una matriz es aquella que, al multiplicarla por la matriz original, da como resultado la matriz identidad. Es útil para resolver sistemas de ecuaciones lineales.

Ejemplo de matriz y su inversa:



La inversa de una matriz[94] es una herramienta matemática muy útil que nos permite "deshacer" transformaciones previas, similar a cómo la división es la operación inversa de la

[94] Para encontrar la matriz inversa de una matriz 2x2, primero calculamos el determinante (multiplicamos los números de la diagonal principal y restamos la multiplicación de la diagonal secundaria: 2×2 - (-1×-1) = 4-1 = 3), luego construimos la matriz adjunta (intercambiamos los elementos de la diagonal principal y cambiamos el signo de los otros), y finalmente dividimos cada elemento de la matriz adjunta entre el determinante, lo que nos da como resultado los elementos de la matriz inversa (por ejemplo, 2/3 = 2÷3).

multiplicación con números normales. Cuando trabajamos con IA, a veces necesitamos retroceder en nuestros cálculos o encontrar valores originales, y la matriz inversa nos permite hacer exactamente eso.

Vamos ahora a ver un ejemplo concreto donde utilizamos las técnicas que aprendimos en las secciones anteriores acerca de cómo se entrena un modelo de IA, pero esta vez utilizando principalmente conceptos de álgebra lineal.

Clasificación de números escritos a mano

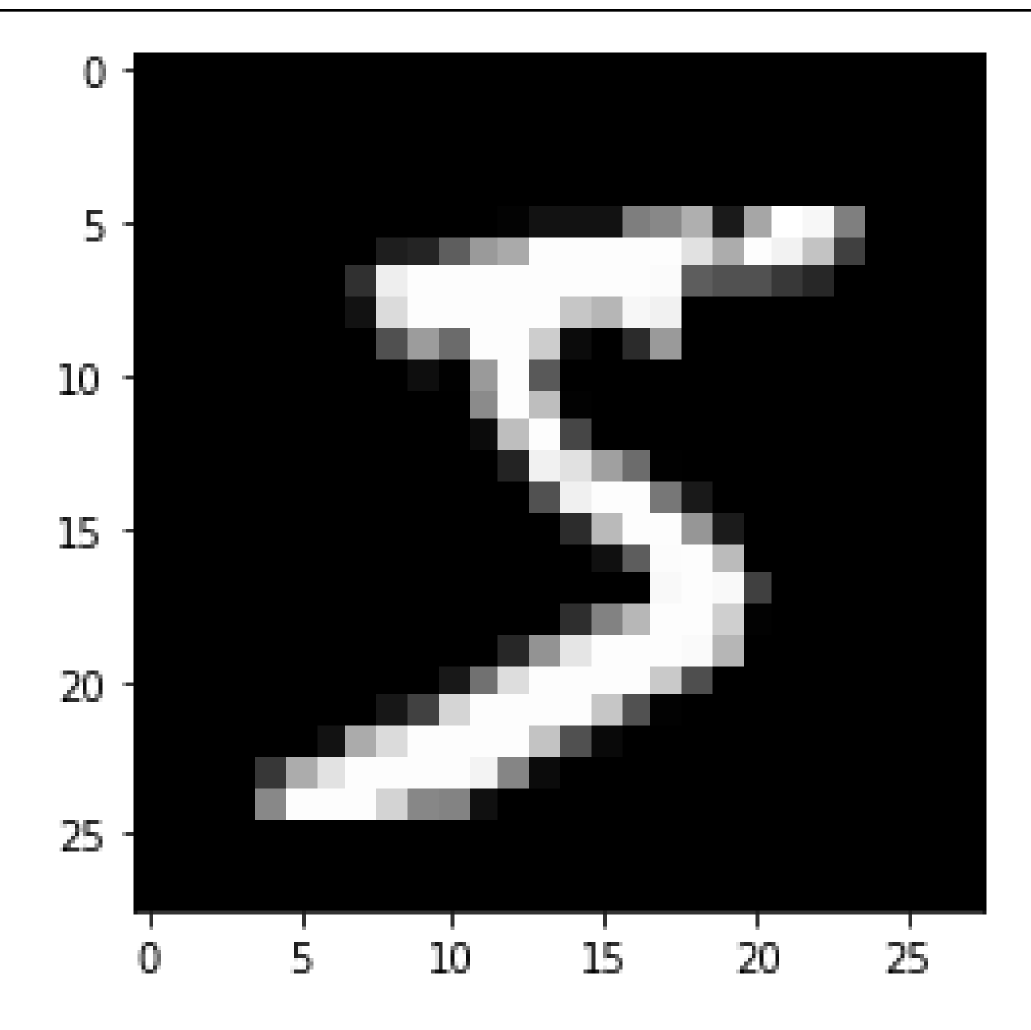
Supongamos que queremos desarrollar una red neuronal para clasificar imágenes de dígitos escritos a mano (como el dataset MNIST[95]).



Cada imagen tiene una resolución de 28x28 píxeles, lo que se traduce en una lista de 784 números (28 x 28) cuando se convierte a una sola fila.

Acá podemos ver un ejemplo de una de ellas:

[95] El conjunto de datos MNIST (Modified National Institute of Standards and Technology) es una base de datos de imágenes digitales de dígitos escritos a mano, ampliamente utilizada en el campo del machine learning y la visión por computadora. Contiene 60,000 imágenes para entrenamiento y 10,000 para pruebas, cada una en escala de grises de 28x28 píxeles. Creado por Yann LeCun, Corinna Cortes y Christopher J.C. Burges, se ha convertido en un estándar de referencia para probar algoritmos de clasificación de imágenes y es frecuentemente utilizado como primer conjunto de datos para introducir conceptos de aprendizaje profundo.



Paso 1: Representación de los datos

En el corazón del procesamiento de imágenes para redes neuronales, está la transformación de información visual en datos numéricos que la computadora pueda procesar. Ya vimos en este libro que cuando tomamos una imagen en escala de grises del conjunto MNIST, cada píxel representa una intensidad de gris que va desde 0 (negro) hasta 255 (blanco)[96]. Esta representación bidimensional de 28x28 píxeles es la forma natural en que visualizamos la imagen del dígito escrito a mano.

En la matriz I que vemos a continuación, cada p_{i,j} representa el valor de intensidad de gris del píxel ubicado en la fila i y columna j de la imagen de arriba. Por ejemplo, p_{1,1} es el valor del píxel en la esquina superior izquierda, mientras que p_{28,28} corresponde al píxel en la esquina inferior derecha. De esta manera, una imagen de un dígito escrito a mano se convierte en una matriz de 28×28 = 784 valores numéricos que la computadora puede procesar y la denotamos de la siguiente manera:

[96] En la práctica, antes de alimentar las imágenes a la red neuronal, cada píxel se normaliza dividiéndolo por 255. Esto significa que en lugar de trabajar con valores enteros entre 0 y 255, la red recibe valores decimales entre 0.0 y 1.0. Por ejemplo, un píxel con valor 127 se convertiría en 127/255 ≈ 0.498, donde 0.0 representa el negro más puro y 1.0 representa el blanco más puro. Esta normalización se realiza porque las redes neuronales tienden a funcionar mejor con números pequeños, ya que ayuda a evitar problemas numéricos durante el entrenamiento y permite que los gradientes se propaguen de manera más efectiva durante el proceso de aprendizaje.

$$I = \begin{bmatrix} p_{1,1} & p_{1,2} & \cdots & p_{1,28} \\ p_{2,1} & p_{2,2} & \cdots & p_{2,28} \\ \vdots & \vdots & \ddots & \vdots \\ p_{28,1} & p_{28,2} & \cdots & p_{28,28} \end{bmatrix}$$

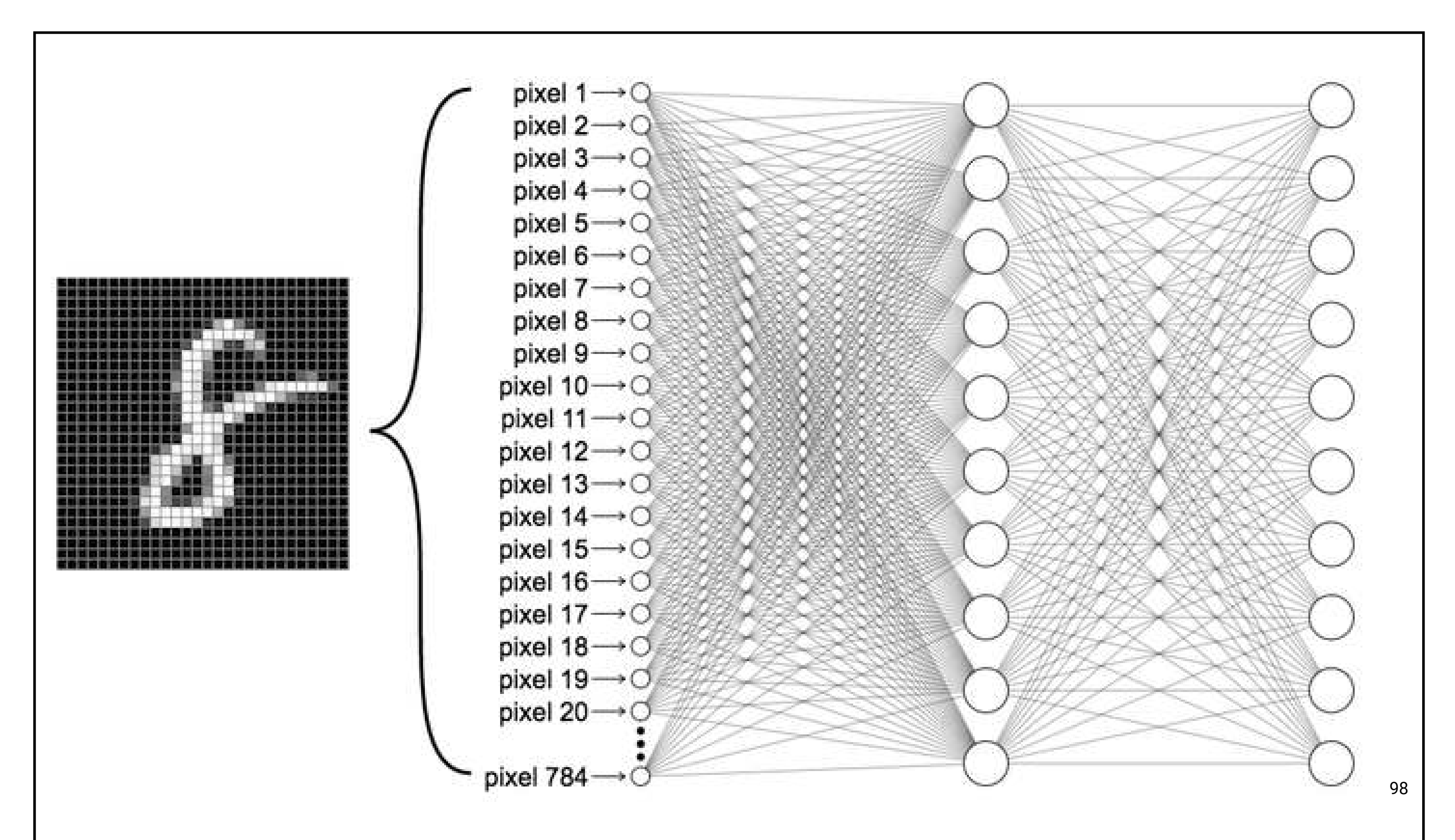
Sin embargo, las redes neuronales necesitan recibir los datos en un formato específico: un vector unidimensional. Por ello, realizamos un proceso llamado "aplanamiento" (flattening), donde convertimos la matriz de 28x28 en una lista única de 784 números[97].

$$x = \begin{bmatrix} p_{1,1} \\ p_{1,2} \\ \vdots \\ p_{1,28} \\ p_{2,1} \\ \vdots \\ p_{28,28} \end{bmatrix}$$

Este proceso es similar a tomar la imagen y "desenrollarla" fila por fila, de izquierda a derecha y de arriba hacia abajo, preservando toda la información original pero en un formato lineal. Esta representación vectorial no solo es una necesidad técnica, sino que también nos permite aplicar todas las herramientas del álgebra lineal a nuestro problema. Los 784 números que componen nuestro vector de entrada pueden verse como coordenadas en un espacio vectorial de 784 dimensiones, donde cada imagen de un dígito escrito a mano es un punto específico en este espacio.

Paso 2: Pesos y Sesgos

[97] La dimensión 784 surge de multiplicar el ancho por el alto de la imagen (28×28 = 784). Este vector unidimensional representa la misma información que la imagen original, pero "desenrollada" en una sola fila, donde cada elemento corresponde a la intensidad de gris de un píxel específico.



[98]

La arquitectura de la red: Imaginemos la red neuronal como un sistema de procesamiento por etapas. La primera etapa (capa de entrada) recibe nuestros 784 números que representan la imagen. La última etapa (capa de salida) nos dará 10 números, donde cada uno representa la probabilidad de que la imagen sea cada uno de los dígitos posibles del 0 al 9. En medio, tenemos una capa intermedia de 128 "neuronas" que ayuda a la red a aprender patrones más complejos. Nuestra red neuronal realiza una serie de transformaciones lineales seguidas de funciones no lineales. En la primera capa, la transformación lineal se puede expresar como:

$$z = Wx + b$$

Donde:

- *x* es nuestra imágen de entrada
- *W* es la matriz de pesos
- *b* es el vector de sesgos
- *z* es el vector de resultados

Los pesos y su organización: Los "pesos" son el corazón del aprendizaje en la red. Para la primera capa, estos pesos se organizan en una matriz de 128×784 ([99]). ¿Por qué estos números? Porque cada una de las 128 neuronas de la capa intermedia necesita conectarse con cada uno de los 784 números de entrada. Si lo visualizamos:

[98] https://ml4a.github.io/ml4a/looking_inside_neural_nets/

[99] La matriz de dimensiones 128×784 significa que tenemos 128 filas (una por cada neurona de la capa oculta) y 784 columnas (una por cada píxel de entrada). Cada fila representa todos los pesos que una única neurona utiliza para procesar la imagen completa.

$$W = \begin{bmatrix} w_{1,1} & w_{1,2} & \cdots & w_{1,784} \\ w_{2,1} & w_{2,2} & \cdots & w_{2,784} \\ \vdots & \vdots & \ddots & \vdots \\ w_{128,1} & w_{128,2} & \cdots & w_{128,784} \end{bmatrix} [128 \times 784]$$

- Cada fila de la matriz representa las conexiones de una neurona
- Cada columna representa cómo un píxel específico influye en todas las neuronas
- El valor en cada posición de la matriz indica qué tan importante es esa conexión específica

El rol del sesgo: El sesgo (también llamado "bias") es como un ajuste fino para cada neurona. Mientras que los pesos multiplican los valores de entrada, el sesgo es un número que se suma al final. En la primera capa, tenemos 128 sesgos (uno por neurona). Este ajuste permite a la red ser más flexible en su aprendizaje.

$$b = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_{128} \end{bmatrix} [128 \times 1]$$

El álgebra lineal hace que todo esto sea computacionalmente eficiente: en lugar de procesar cada conexión una por una (¡que serían 128×784 operaciones individuales!), podemos usar una simple multiplicación de matrices para procesar todas las conexiones de una vez. Esta eficiencia es crucial cuando estamos trabajando con miles de imágenes durante el entrenamiento.

Antes de continuar, repasemos lo que vimos hasta ahora. Comenzamos con una imagen de 28x28 píxeles de un número escrito a mano, donde cada píxel es un valor entre 0 (negro) y 255 (blanco). Esta imagen la "desenrollamos" en una lista de 784 números, que es la forma en que la red neuronal puede procesarla. Luego, presentamos la primera capa de nuestra red, que consiste en 128 neuronas, cada una con sus propios pesos y sesgo. Cada neurona examina todos los píxeles de la imagen, pero da diferente importancia a cada uno según sus pesos (como si cada neurona tuviera su propia "opinión" sobre qué partes de la imagen son más importantes). Los pesos y sesgos son los parámetros que la red ajustará durante el aprendizaje para mejorar su capacidad de reconocimiento. Todo esto se realiza mediante operaciones de álgebra lineal, específicamente multiplicaciones de matrices, que permiten procesar toda esta información de manera eficiente.

Paso 3: Cálculo de la Activación

Después de esta transformación lineal, cada neurona aplica una función especial llamada ReLU (Rectified Linear Unit).

$$h_i = \text{ReLU}(z_i) = \max(0, z_i)$$

Esta función es sorprendentemente simple: si el número que recibe es negativo, lo convierte en cero; si es

positivo, lo deja tal cual[100]. Aunque parece una operación sencilla, es crucial porque introduce no-linealidad en el sistema, permitiendo que la red aprenda patrones más complejos. Sin esta función, toda la red sería equivalente a una única transformación lineal, lo que limitaría severamente su capacidad de aprendizaje.

$$h = \mathrm{ReLU}(z) = \mathrm{ReLU}(Wx + b)$$

Este proceso se repite simultáneamente en las 128 neuronas de nuestra capa intermedia, cada una con sus propios pesos y sesgo únicos. Gracias al álgebra lineal, podemos realizar todas estas operaciones de manera muy eficiente usando multiplicación de matrices, en lugar de hacer cada cálculo individualmente. Al final, cada neurona produce un número que representa qué tan fuertemente detectó ciertos patrones en la imagen. Algunas neuronas podrían especializarse en detectar bordes, otras en detectar curvas, y así sucesivamente, creando una representación rica y multidimensional de la imagen original.

Paso 4: Capa de Salida

En la capa de salida ocurre la transformación final de nuestra red, donde las 128 características detectadas por la capa intermedia se convierten en predicciones sobre qué dígito representa la imagen. Acá, cada una de las 10 neuronas de salida (una por cada posible dígito del 0 al 9) se conecta con todas las características mediante una nueva matriz de pesos de 10×128[101]. Es como si cada neurona de salida tuviera su propia "receta" para combinar las características y decidir si la imagen corresponde a su dígito asignado.

$$y = \mathrm{softmax}(Vh + c)$$

Esta fórmula representa el primer gran procesamiento que hace nuestra red neuronal. Podemos pensarla como si cada una de las 128 neuronas fuera un detector de patrones con sus propias "preferencias" (los pesos W) que examina la imagen completa (x). Cada neurona multiplica cada píxel por su peso correspondiente (Wx) y añade su propio ajuste fino (b), como un chef que combina ingredientes según su receta personal y ajusta el punto de sal. Finalmente, ReLU actúa como un filtro que dice "si encontraste algo relevante (número positivo), mantenelo; si no (número negativo), ignoralo completamente (convertilo a cero)".

Donde:

- *h* es el vector de características de la capa intermedia que generamos en el paso 3
- *V* es la matriz de pesos de la capa de salida
- *c* es el vector de sesgos de la capa de salida
- *y* es el vector de probabilidades resultante

[100] La función ReLU (Rectified Linear Unit) es una de las funciones de activación más populares en redes neuronales por su simplicidad: si el número es positivo, lo deja pasar tal cual; si es negativo, lo convierte en cero. Por ejemplo, si recibe los valores [-2, 3, -1, 4], devuelve [0, 3, 0, 4]. Esta simplicidad la hace computacionalmente eficiente y ayuda a que las redes neuronales aprendan mejor.

[101] La matriz 10×128 conecta las 128 neuronas de la capa oculta con las 10 neuronas de salida. Cada fila representa los pesos que una neurona de salida (correspondiente a un dígito del 0 al 9) usa para combinar las características detectadas por la capa oculta.

$$V = \begin{bmatrix} v_{1,1} & v_{1,2} & \cdots & v_{1,128} \\ v_{2,1} & v_{2,2} & \cdots & v_{2,128} \\ \vdots & \vdots & \ddots & \vdots \\ v_{10,1} & v_{10,2} & \cdots & v_{10,128} \end{bmatrix} [10 \times 128]$$

$$c = \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_{10} \end{bmatrix} [10 \times 1]$$

Después de realizar esta transformación lineal (multiplicación de matrices y suma de sesgos), aplicamos una función especial llamada softmax, que es diferente a la ReLU que usamos antes. La función softmax toma los 10 números resultantes y los convierte en probabilidades que suman 1, donde valores más altos indican mayor confianza en la predicción. Por ejemplo, si la red está muy segura de que una imagen es un 7, la neurona correspondiente al 7 podría tener una probabilidad de 0.85 (85% de confianza), mientras que las otras neuronas se repartirían el 0.15 restante.

El álgebra lineal vuelve a ser fundamental en este paso porque nos permite realizar estas transformaciones de manera eficiente mediante una única multiplicación matricial, en lugar de procesar las 1,280 conexiones (10×128) una por una. Esta eficiencia es crucial cuando procesamos miles de imágenes, tanto durante el entrenamiento como en la predicción[102]. Además, la función softmax nos proporciona una interpretación probabilística clara de las predicciones de la red, permitiéndonos entender no sólo qué dígito cree la red que es, sino también cuán segura está de su predicción.

Podemos expresar todo el proceso desde la imágen hasta la predicción final como:

$$y = \mathrm{softmax}(V \cdot \mathrm{ReLU}(Wx + b) + c)$$

Esta fórmula, aunque parece compleja, describe el viaje completo de nuestra imagen hasta obtener la predicción final. Podemos pensarla como una receta que sigue varios pasos: primero, la imagen (x) se procesa con la primera capa de la red (W·x + b), como si cada neurona examinara la imagen con su propia "lupa". Luego, ReLU actúa como un filtro que decide qué información es importante mantener (los números positivos) y cuál descartar (convirtiendo los negativos a cero). Finalmente, la última capa (V) combina toda esta información filtrada y softmax la convierte en porcentajes de confianza para cada dígito posible, similar a cuando un estudiante reparte 100 puntos entre diferentes respuestas según qué tan seguro está de cada una.

Paso 5: Resultado Final y Aplicación Práctica

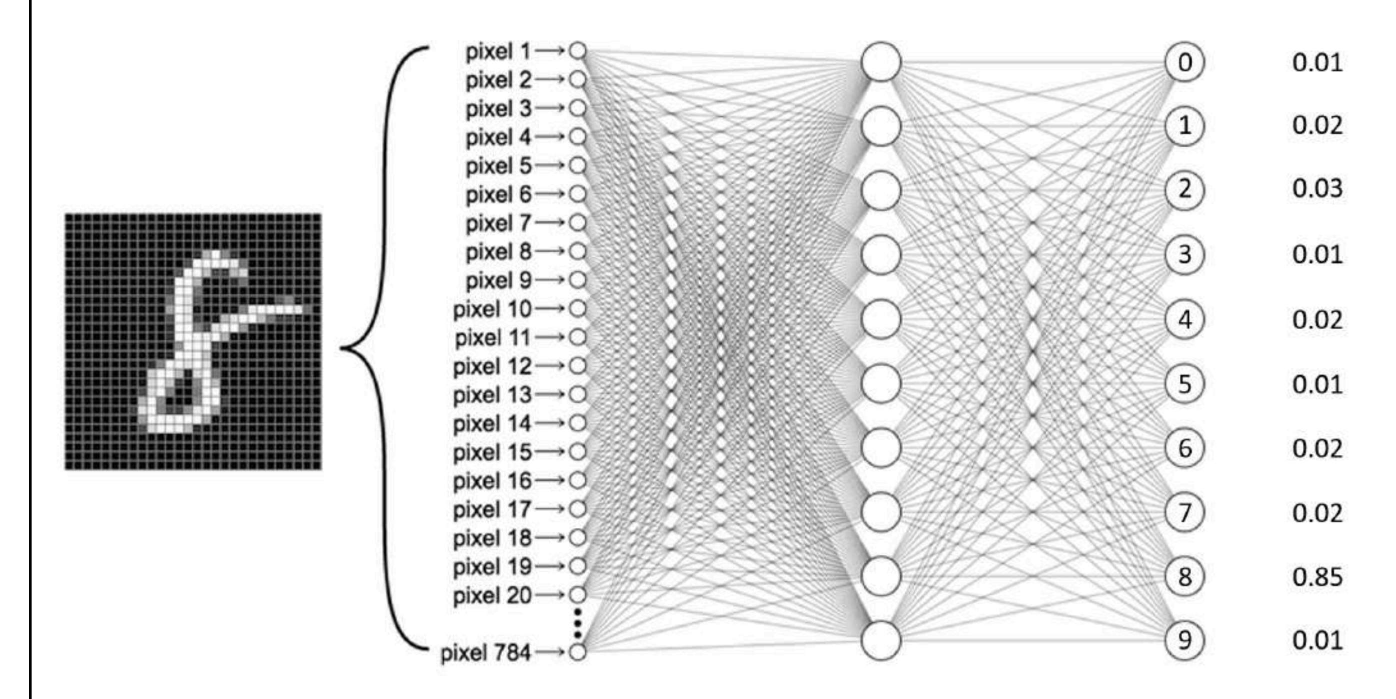
Después de todas estas transformaciones, nuestra red neuronal produce un vector de 10 números entre

[102] En la práctica, las redes neuronales no procesan las imágenes una por una, sino que trabajan con "mini-batches" o lotes pequeños de imágenes simultáneamente (por ejemplo, 32 o 64 imágenes a la vez). Esto significa que en lugar de tener una única imagen de entrada de dimensión 784, tenemos una matriz de dimensión 32×784, lo que permite aprovechar el paralelismo del hardware moderno y acelerar significativamente el entrenamiento y la inferencia.

0.00 y 1.00, donde cada número representa la probabilidad de que la imagen de entrada sea cada uno de los dígitos posibles.

Por ejemplo, podría dar como resultado: [0.01, 0.02, 0.03, 0.01, 0.02, 0.01, 0.02, 0.02, **0.85**, 0.01]

Este resultado nos indicaría que la red está 85% segura de que el dígito en la imagen es un "8" (considerando que la lista comienza desde el cero).



Acabamos de ver cómo una red neuronal, en su núcleo más básico, no es más que una serie de transformaciones de álgebra lineal cuidadosamente organizadas. Todo comienza con la representación de una imagen como un vector (álgebra lineal) de 784 números. Luego, mediante multiplicaciones de matrices (álgebra lineal) y la adición de vectores de sesgos (más álgebra lineal), la información fluye a través de 128 neuronas intermedias. Aunque introducimos elementos no lineales como la función ReLU para dar al sistema más capacidad de aprendizaje, la mayor parte del procesamiento sigue siendo álgebra lineal pura. Finalmente, otra multiplicación de matrices nos lleva a las 10 neuronas de salida, donde la función softmax nos da las probabilidades finales. Es sorprendente cómo algo tan complejo como el reconocimiento de dígitos escritos a mano puede reducirse, en esencia, a una serie de operaciones de álgebra lineal bien diseñadas y estratégicamente combinadas con unas pocas funciones no lineales.

Conclusión

Este ejemplo ilustra cómo el álgebra lineal no es solo una herramienta matemática abstracta, sino el fundamento operativo de las redes neuronales modernas. Cada operación que se ha descrito, desde la representación inicial de la imagen hasta la predicción final, depende de conceptos fundamentales del álgebra lineal:

- Vectorización de datos
- Multiplicación de matrices
- Transformaciones lineales
- Espacios vectoriales

La belleza de este sistema radica en que, a pesar de su aparente complejidad, se basa en operaciones

matemáticas relativamente simples que, cuando se combinan, permiten a la red aprender patrones increíblemente complejos. Esta es la razón por la que el álgebra lineal se ha convertido en una herramienta indispensable en el campo de la inteligencia artificial y el *machine learning*.

Este ejemplo muestra cómo usamos álgebra lineal para procesar datos en una red neuronal. Aunque puede sonar complejo, en el fondo, todo se reduce a una cadena de multiplicaciones de matrices y transformaciones no lineales que, paso a paso, convierten un dato de entrada en una predicción. Esta es la esencia de cómo funcionan muchas aplicaciones de inteligencia artificial.

De todas maneras, mientras que el álgebra lineal nos proporciona la estructura básica de la red y define cómo fluye la información a través de ella, necesitamos el *cálculo diferencial* para que la red pueda "aprender". El proceso de optimización utiliza derivadas parciales para calcular el *gradiente*, que nos indica cómo cada peso en nuestras matrices afecta al error final. Cada vez que la red hace una predicción incorrecta, el cálculo diferencial nos permite determinar exactamente cómo cada peso contribuyó a ese error y, lo más importante, en qué dirección deberíamos ajustar cada peso para reducir ese error. Es similar a estar en una montaña (donde la altura representa el error) y usar el cálculo para determinar la dirección de la pendiente más empinada en cada punto, permitiéndonos descender gradualmente hacia el valle (el punto de menor error). Esta aplicación del cálculo, conocida como *descenso de gradiente*, es lo que permite que nuestras redes neuronales aprendan de sus errores y mejoren con el tiempo y es lo que vamosa repasar ahora.

¿Por qué necesitamos el cálculo?

El *cálculo* es una herramienta matemática fundamental en la inteligencia artificial. Proporciona los métodos necesarios para optimizar modelos y analizar cómo estos cambian en respuesta a diferentes variables. Las *derivadas* y las *integrales*, componentes centrales del cálculo, son esenciales en el *machine learning* para *ajustar* parámetros de los modelos, minimizar el error utilizando funciones de pérdida y evaluar el desempeño de los algoritmos.

Vamos a ver un ejemplo concreto, pero antes, dos conceptos importantes:

Derivadas

La derivada de una función en un punto se define como el límite del cociente entre el cambio en "y" (altura) y el cambio en "x" (avance horizontal) cuando este avance se hace infinitamente pequeño. En términos más simples, la derivada en un punto nos indica la **tasa de cambio instantánea** de la función en ese momento específico, o dicho de otra manera, la **pendiente** de la recta tangente a la curva en ese punto. Es similar a cómo el velocímetro de un coche nos muestra la velocidad instantánea: no nos dice la velocidad promedio del viaje, sino exactamente qué tan rápido estamos yendo en ese preciso momento.

1. Medición de cambios

Imagina que estás conduciendo un coche. La derivada es como el velocímetro de tu coche.

Analogía: Si el velocímetro marca 60 km/h, significa que tu posición está cambiando a una velocidad de 60 kilómetros por cada hora.

En Machine Learning, la derivada nos dice qué tan rápido cambia el error de nuestro modelo cuando ajustamos un parámetro.



Matemáticamente, se obtiene como el límite del cociente de las diferencias (también llamado cociente incremental) cuando el incremento de la variable independiente tiende a cero.

2. Ajuste de parámetros

Piensa en sintonizar una radio para encontrar tu estación favorita.

Analogía: Giras el dial lentamente. Si el sonido mejora, sigues en esa dirección. Si empeora, giras en la dirección opuesta.

En Machine Learning, usamos derivadas para saber en qué dirección "girar los diales" (ajustar los parámetros) para mejorar nuestro modelo.



3. Minimización de errores

Imagina que estás tratando de encontrar el punto más bajo en un valle con los ojos vendados.

Analogía: Das pequeños pasos y sientes si el suelo baja o sube. Siempre te mueves hacia donde el suelo baja más rápidamente.

En Machine Learning, las derivadas nos ayudan a encontrar el "valle" donde el error de nuestro modelo es el más bajo posible.



Gradiente y Descenso de Gradiente

Relacionado con el ejemplo anterior, el descenso de gradiente es un proceso que se asemeja al de bajar una colina para encontrar el punto más bajo. Imaginá que estás en una colina con los ojos vendados y querés llegar al valle. En cada paso, tocás el suelo a tu alrededor para sentir la pendiente y das un paso en la dirección donde el terreno baja más. Esta "pendiente" que sentís es matemáticamente equivalente a la derivada de la función en ese punto, que nos indica la tasa de cambio o inclinación del terreno. Si la derivada es positiva, el terreno sube; si es negativa, baja. El gradiente (que generaliza la derivada a múltiples dimensiones) nos indica la dirección de máximo crecimiento, por lo que si te movés en la dirección opuesta te va a llevar hacia el valle. Repetís este proceso muchas veces, ajustando la dirección y el tamaño de tus pasos según lo empinado del terreno, que viene dado por la magnitud de la derivada. En *machine learning*, la "colina" es una función de error del modelo, y el "valle" es la configuración óptima de los parámetros del modelo que minimiza este error. El algoritmo ajusta iterativamente los parámetros, siguiendo la dirección de máximo descenso indicada por el gradiente negativo de la función de error, hasta encontrar el mínimo o un punto suficientemente bajo.

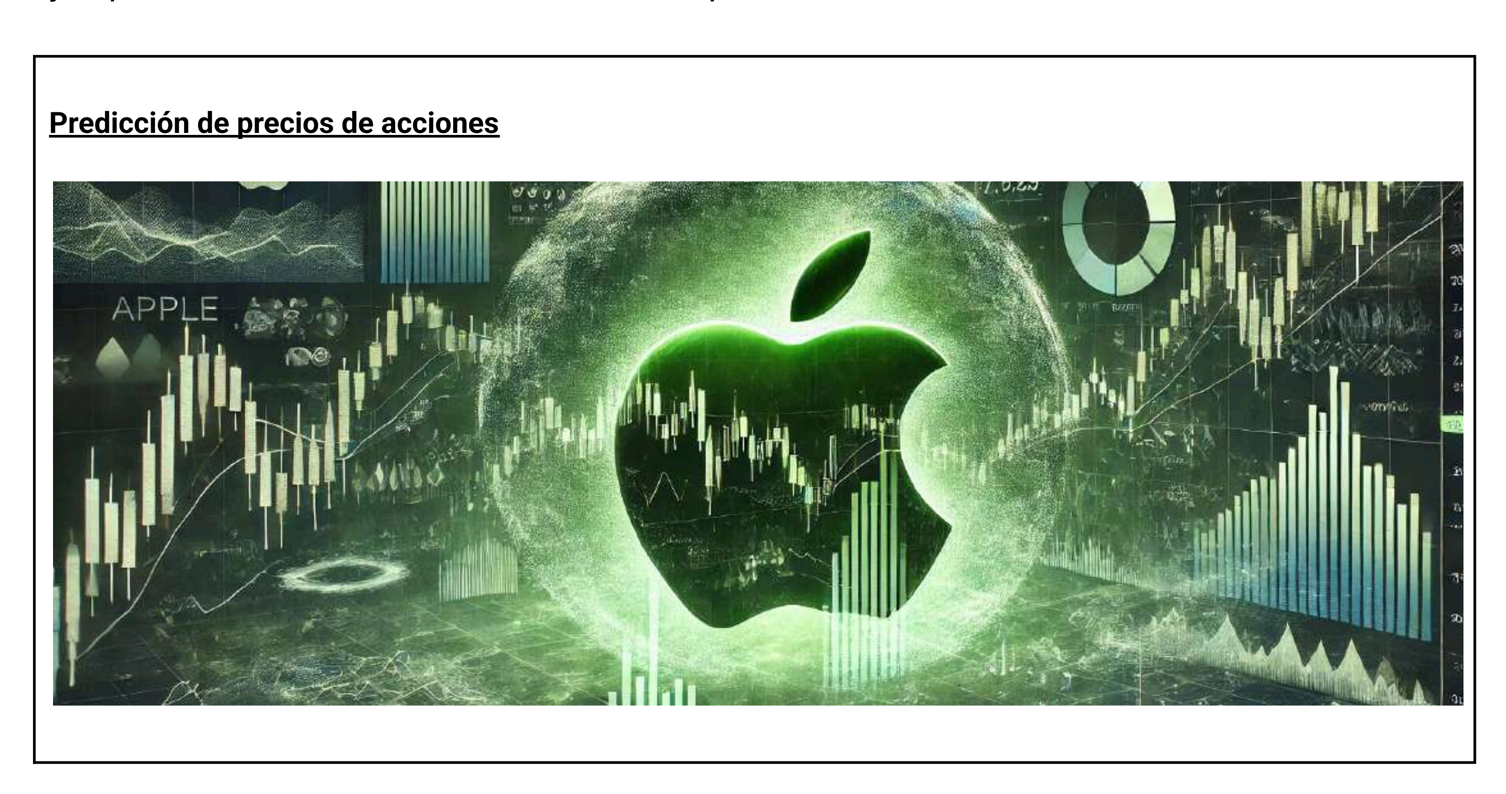
Descenso de Gradiente en la función f(x) = x^2 + 4

Función de costo f(x) = x^2 + 4

Trayectoria del descenso de gradiente

Con estos dos conceptos (derivadas y descenso de gradiente) presentados, vamos a ver un ejemplo de cómo el *cálculo* es fundamental para la IA.

Predicción de precios de acciones



Imaginemos un caso práctico: estamos entrenando una red neuronal para predecir el **precio de la acción de Apple mañana**. Este proceso se puede entender en tres pasos fundamentales:

1. Primero, elegimos un *modelo paramétrico*[103] (como una red neuronal o una regresión lineal). Estos modelos tienen parámetros ajustables que determinan su comportamiento - similar a las perillas de una radio.
2. Después, definimos una *función de pérdida* que mide qué tan equivocadas están nuestras predicciones. Por ejemplo, podemos calcular la diferencia promedio entre los precios que predice nuestro modelo y los precios reales históricos de la acción. Cuanto mayor sea esta diferencia, mayor será nuestra "pérdida" o error.
3. Finalmente, nuestro objetivo se convierte en un problema matemático de optimización: necesitamos encontrar los valores específicos de los parámetros[104] que minimicen esta función de pérdida. En otras palabras, buscamos ajustar las "perillas" de nuestro modelo hasta que sus predicciones se acerquen lo más posible a la realidad.

El cálculo matemático, especialmente el concepto de derivadas que vimos anteriormente, es la herramienta fundamental que nos permite resolver este problema de optimización de manera eficiente.

Selección del modelo paramétrico

Supongamos que ya elegimos un modelo de regresión lineal, uno de los modelos paramétricos más simples, para nuestra predicción. En este caso, nuestro modelo sería una línea recta definida por dos parámetros: la pendiente (β_1) y la intersección con el eje Y (β_0). Cada combinación de estos parámetros generará diferentes predicciones para el precio de la acción. Por ejemplo, una línea con β_1 = 1.5 y β_0 = 100 predecirá precios diferentes a una línea con β_1 = 0.8 y β_0 = 150. Nuestra tarea es encontrar los valores específicos de β_1 y β_0 que generen las predicciones más precisas, es decir, que minimicen la diferencia entre los precios predichos y los precios reales históricos de la acción.

---

[103] Un modelo paramétrico es una representación matemática que describe un fenómeno o proceso mediante un conjunto fijo de parámetros ajustables. Por ejemplo, una línea recta es un modelo paramétrico definido por dos parámetros: la pendiente y la intersección con el eje Y. La característica distintiva de estos modelos es que su estructura básica está predefinida y lo que aprendemos son los valores óptimos de sus parámetros a partir de los datos, en contraste con los modelos no paramétricos que pueden adaptar su estructura de forma más flexible.

[104] Un parámetro es una variable en un modelo matemático o estadístico que influye en el comportamiento y los resultados del modelo. En el contexto del machine learning, los parámetros son valores que el modelo aprende durante el proceso de entrenamiento a partir de los datos. Estos valores determinan cómo el modelo procesa la información de entrada para generar predicciones o resultados. Por ejemplo, en una regresión lineal, los coeficientes de las variables y el término independiente son parámetros. En una red neuronal, los pesos de las conexiones entre neuronas y los sesgos son parámetros. El ajuste de estos parámetros durante el entrenamiento permite al modelo adaptarse a los patrones en los datos y realizar predicciones más precisas. Para ilustrar esto, consideremos una regresión lineal simple: y = mx + b, donde 'y' es la variable que queremos predecir, 'x' es la variable independiente, 'm' es la pendiente y 'b' es el intercepto. En este caso, 'm' y 'b' son los parámetros del modelo. Si estamos prediciendo el precio de una casa (y) basándonos en su tamaño en metros cuadrados (x), el modelo podría aprender que m = 1000 y b = 50000. Esto significaría que por cada metro cuadrado adicional, el precio aumenta en 1000 euros, y una casa de 0 metros cuadrados tendría un precio base de 50000 euros. Estos parámetros (m y b) se ajustan durante el entrenamiento para minimizar la diferencia entre las predicciones del modelo y los precios reales de las casas en nuestros datos.

Definición de la función de pérdida

Paso siguiente, definimos una función de pérdida que mida el error entre las predicciones del modelo y los valores reales. Una función de pérdida común es la función de *error cuadrático medio* (MSE por *mean squared error* en inglés), que se calcula como:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

Para facilitar la lectura en este libro la convertiremos a: **MSE** = (1/n) * sum((y_i - y_pred_i)^2)[105]

Donde:

- ***MSE*** es el error cuadrático medio.
- ***n*** es el número total de ejemplos con los que estamos evaluando la calidad del modelo
- ***y_i*** son cada uno de los valores reales.
- ***y_pred_i*** son cada una de las predicciones del modelo.

Si observás bien, cuanto más se parezcan los valores reales (y_i) a las predicciones del modelo (y_pred_i), más pequeño será el error MSE y por lo tanto mejor será nuestro modelo. El *cálculo* es importante acá porque necesitamos minimizar esta función de pérdida para mejorar el desempeño del modelo.

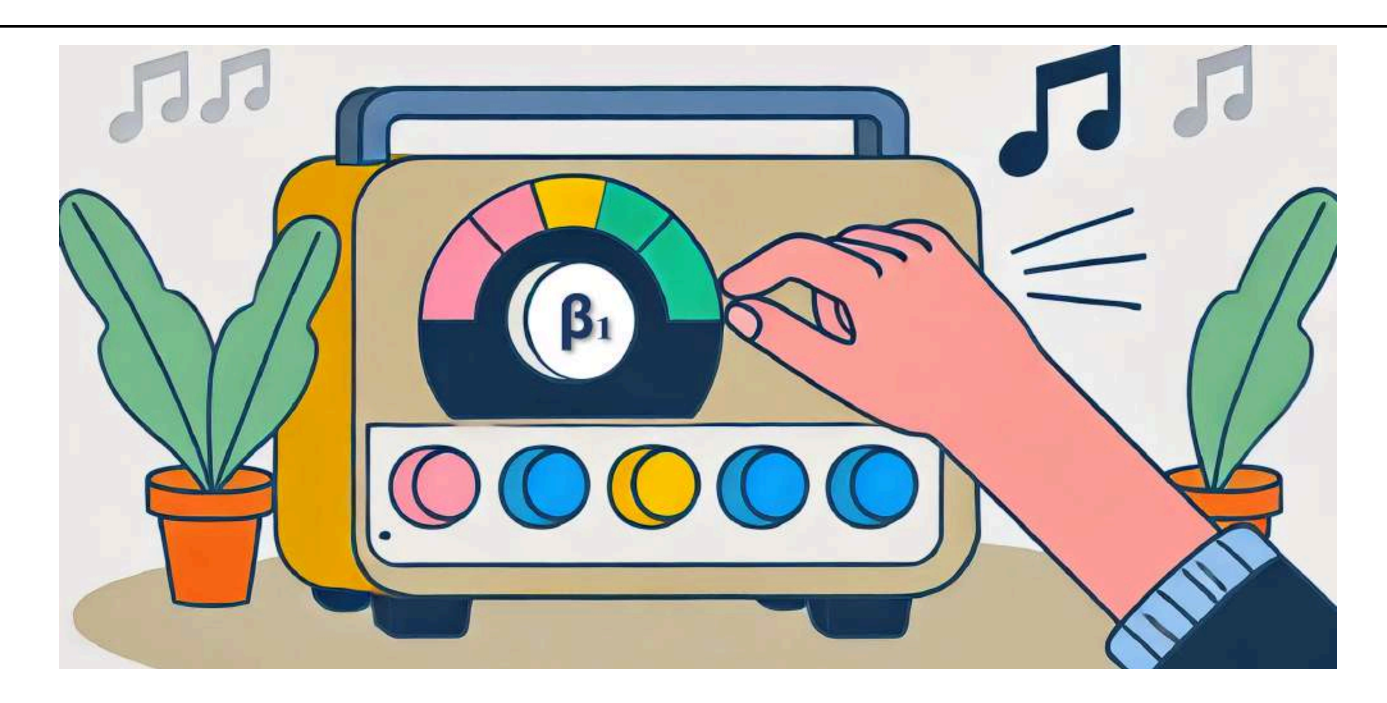
Cálculo del gradiente

Para minimizar la función de pérdida ya vimos la analogía de *descender por el valle* y *ajustar los parámetros haciendo sintonía fina*, entonces técnicamente calculamos el *gradiente* de la función de pérdida con respecto a cada parámetro del modelo. El *gradiente* es un vector de *derivadas parciales* que indica cómo cambia la función de pérdida con respecto a cada parámetro.

Consideremos que un parámetro de nuestro modelo que estamos ajustando sea $\boldsymbol{\beta_1}$ (la pendiente de nuestro modelo de regresión lineal). La derivada parcial de la función de pérdida con respecto a $\boldsymbol{\beta_1}$ se calcularía sumando el producto del error (y_i - y_pred_i) y la derivada de la predicción con respecto a $\boldsymbol{\beta_1}$, para todos los ejemplos. En esencia lo que estamos haciendo es evaluar qué pasa en un entorno muy cercano a $\boldsymbol{\beta_1}$ y en qué sentido podríamos mover $\boldsymbol{\beta_1}$ para estar un poco mejor, es decir, la *sintonía fina* que comentábamos antes.[106]

[105] y_i es uno de los valores reales mientras que y_pred_i es una de las predicciones que el modelo hace. La resta de ambas nos da el error del modelo. Elevar ese error al cuadrado no solamente convierte el error en un numero positivo (por que menos por menos es más) sino que además magnifica los errores grandes, es decir, el modelo penaliza más fuertemente a los grandes errores. Finalmente, sumamos esos errores cuadráticos para cada uno de los valores de nuestro dataset de entrenamiento y lo dividimos por la cantidad de estos datos, para obtener finalmente el error cuadrático medio.

[106] Concretamente en nuestro modelo de regresión lineal, el gradiente de nuestra función de pérdida se calcula tomando las derivadas parciales con respecto a cada uno de nuestros parámetros (β_0 y β_1). Matemáticamente, si nuestra función de pérdida es el Error Cuadrático Medio (MSE), que suma los cuadrados de las diferencias entre las predicciones ($\hat{y}_i = \beta_0 + \beta_1 x_i$) y los valores reales ($y_i$), el gradiente será un vector con dos componentes: $\partial MSE/\partial\beta_0$ y $\partial MSE/\partial\beta_1$. La primera componente nos indica cómo cambia el error cuando modificamos la intersección, mientras que la segunda nos muestra cómo varía el error al ajustar la pendiente.



El cálculo de esta derivada nos dice entonces cómo ajustar β_1 para reducir el error.

Descenso de Gradiente

Utilizamos el método de *descenso de gradiente* visto anteriormente para actualizar los parámetros del modelo. Esto implica mover los parámetros en la dirección opuesta al *gradiente* para reducir el resultado evaluado de la función de pérdida. La actualización del parámetro β_1 se realiza de la siguiente manera:

$\beta_1 = \beta_1$ - (learning_rate * gradiente)

Donde:

- β_1 es el parámetro que estamos ajustando.
- ***learning_rate*** (*tasa de aprendizaje*) es un valor pequeño que determina el tamaño del paso que damos en la dirección del gradiente. Esto quiere decir que aunque sepamos hacia donde queremos ajustar β_1, podemos elegir ser prudentes y hacerlo de a poco. Qué tan prudentes queremos ser dependerá del valor que le demos a *learning_rate*. El riesgo de un *learning_rate* muy alto es pasarnos de donde queríamos llegar y así generar que nuestro modelo nunca encuentre su configuración óptima[107].
- ***gradiente*** es la derivada parcial que calculamos anteriormente.

Imaginemos que nuestro parámetro β_1 actual es 5.0, nuestra **tasa de aprendizaje** (**learning_rate**) es 0.1, y el ***gradiente*** calculado es 2.0. Aplicando la fórmula:

[107] A este fenómeno se lo conoce como *overshooting*. El overshooting o sobreajuste en el descenso de gradiente ocurre cuando el tamaño del paso (*learning rate*) es demasiado grande, causando que el algoritmo "salte" por encima del mínimo óptimo. Podemos imaginarlo como un auto que frena demasiado tarde y se pasa de su destino: aunque sabe la dirección correcta, la velocidad (análoga al *learning rate*) es tan alta que no puede detenerse en el punto deseado. En el contexto del *machine learning*, esto se manifiesta como oscilaciones en los valores de los parámetros del modelo, donde estos fluctúan de un extremo a otro sin converger al valor óptimo, lo que puede resultar en un entrenamiento inestable o incluso en la divergencia completa del modelo.

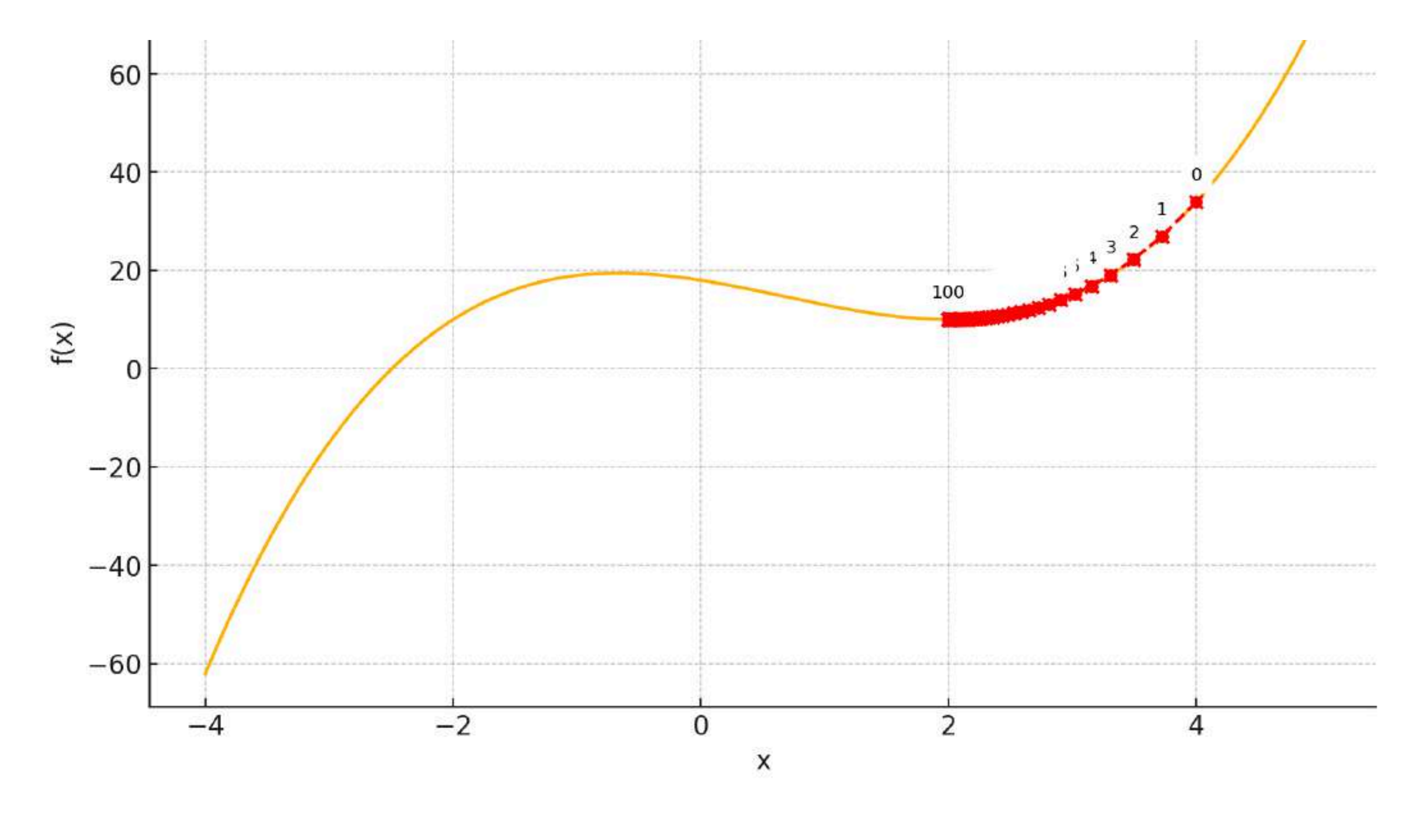
β_1 = 5.0 - (0.1 * 2.0)

= 5.0 - 0.2

= 4.8.

Esto significa que nuestro parámetro β_1 se ha actualizado de 5.0 a 4.8, moviéndose un pequeño paso de 0.2 en la dirección opuesta al gradiente con el objetivo de minimizar el error.

Cuando el proceso de *descenso de gradiente* haya completado luego de una gran cantidad de iteraciones (repeticiones del proceso anterior) llegará un punto donde el valor del *gradiente* será 0.0 y por lo tanto, siguiendo la ecuación anterior, todo el segundo término (el que va entre paréntesis) será igual a cero y nos quedará $\beta_1 = \beta_1 - 0$, que es lo mismo que decir que β_1 no cambiará más, por que ya no consigue reducir más la pérdida, entonces el modelo dejará de *entrenarse* y podremos proceder a evaluar qué tan bueno es ese modelo prediciendo el precio de la acción de Apple de mañana. ¡Ese será nuestro modelo entrenado y todo gracias al cálculo!



Nota: en este libro vimos muchos algoritmos diferentes, pero quiero que recuerdes que este ejemplo es agnóstico al algoritmo. Mientras sea un modelo paramétrico (en nuestro ejemplo una regresión lineal, pero también aplica en un *support vectors machine* o una red neuronal) el proceso es el mismo: optimizar sus parámetros mediante un proceso de *descenso de gradiente* de la función de pérdida. Lo que cambia de algoritmo a algoritmo es cuáles son esos parámetros *entrenables*. En nuestro ejemplo asumimos un único parámetro β_1, pero sabemos que la regresión lineal también tiene otro parámetro entrenable que es β_0. En la realidad podemos tener algo tan parecido de sencillo con un par de parámetros o algo tan complejo como un LLM con **billones** (si, billones con b) de parámetros. Que no te asuste, el procedimiento es el mismo, solo que billones de veces más costoso.

Este ejemplo muestra cómo el cálculo, y en particular las derivadas y el descenso de gradiente se utilizan para optimizar/entrenar los modelos de machine learning. Al calcular la pendiente de la función de pérdida y ajustar los parámetros en consecuencia, podemos minimizar el error y mejorar el desempeño del modelo. Aunque puede sonar complejo, la idea básica es que utilizamos las matemáticas para entender cómo cambiar los parámetros del modelo para que haga mejores predicciones. Esta es la esencia de cómo funcionan muchas aplicaciones de inteligencia artificial.

Sin embargo, optimizar los parámetros de nuestro modelo es solo una parte de la historia. En el mundo real, los datos raramente son perfectos y las relaciones entre variables no son determinísticas. Por ejemplo, el precio de una acción no solo depende de sus valores históricos, sino que está sujeto a numerosas fuentes de incertidumbre: eventos geopolíticos, decisiones empresariales, sentimiento del mercado, entre otros. Es acá donde la probabilidad y la estadística se vuelven fundamentales: necesitamos herramientas que nos permitan modelar y cuantificar esta incertidumbre para tomar mejores decisiones.

Pero…

¿Por qué necesitamos la probabilidad y la estadística?

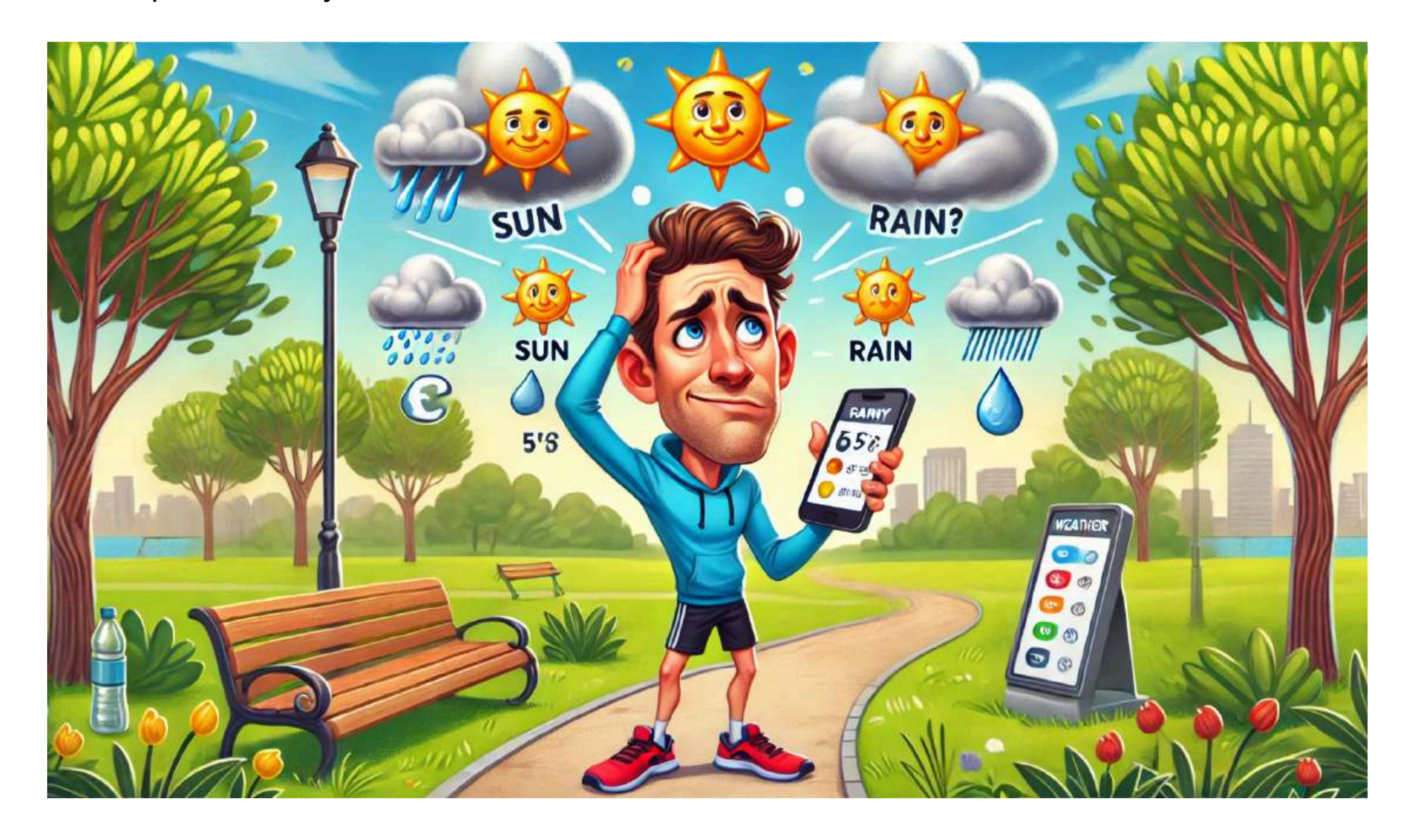


La probabilidad y la estadística son los fundamentos matemáticos que permiten a la inteligencia artificial tomar decisiones y aprender de los datos. Así como nuestro cerebro aprende de patrones y experiencias pasadas, la IA utiliza métodos probabilísticos para identificar tendencias, hacer predicciones y manejar la incertidumbre del mundo real. Por ejemplo, cuando un *modelo* reconoce caras en una fotografía o predice la siguiente palabra que vas a escribir en un mensaje de texto, está utilizando estadística para calcular la probabilidad de cada posible palabra y elegir la más adecuada. Sin estas herramientas matemáticas, la IA sería incapaz de procesar la gran cantidad de datos que recibe y transformarlos en decisiones útiles, convirtiendo así a la probabilidad y la estadística en un concepto fundamental que permite a las máquinas "pensar" y "aprender".

<u>Distribuciones de Probabilidad</u>

Sería imposible cubrir en este libro la enorme extensión de conceptos que abarcan la probabilidad y la estadística. Por esto mismo, nos vamos a enfocar en una herramienta fundamental, las distribuciones de probabilidad, que son importantísimas para entender la incertidumbre en los datos y constituyen la base de cómo los sistemas de IA aprenden y toman decisiones.

Supongamos que estás desarrollando una aplicación con IA para recomendar el mejor horario para hacer ejercicio al aire libre.

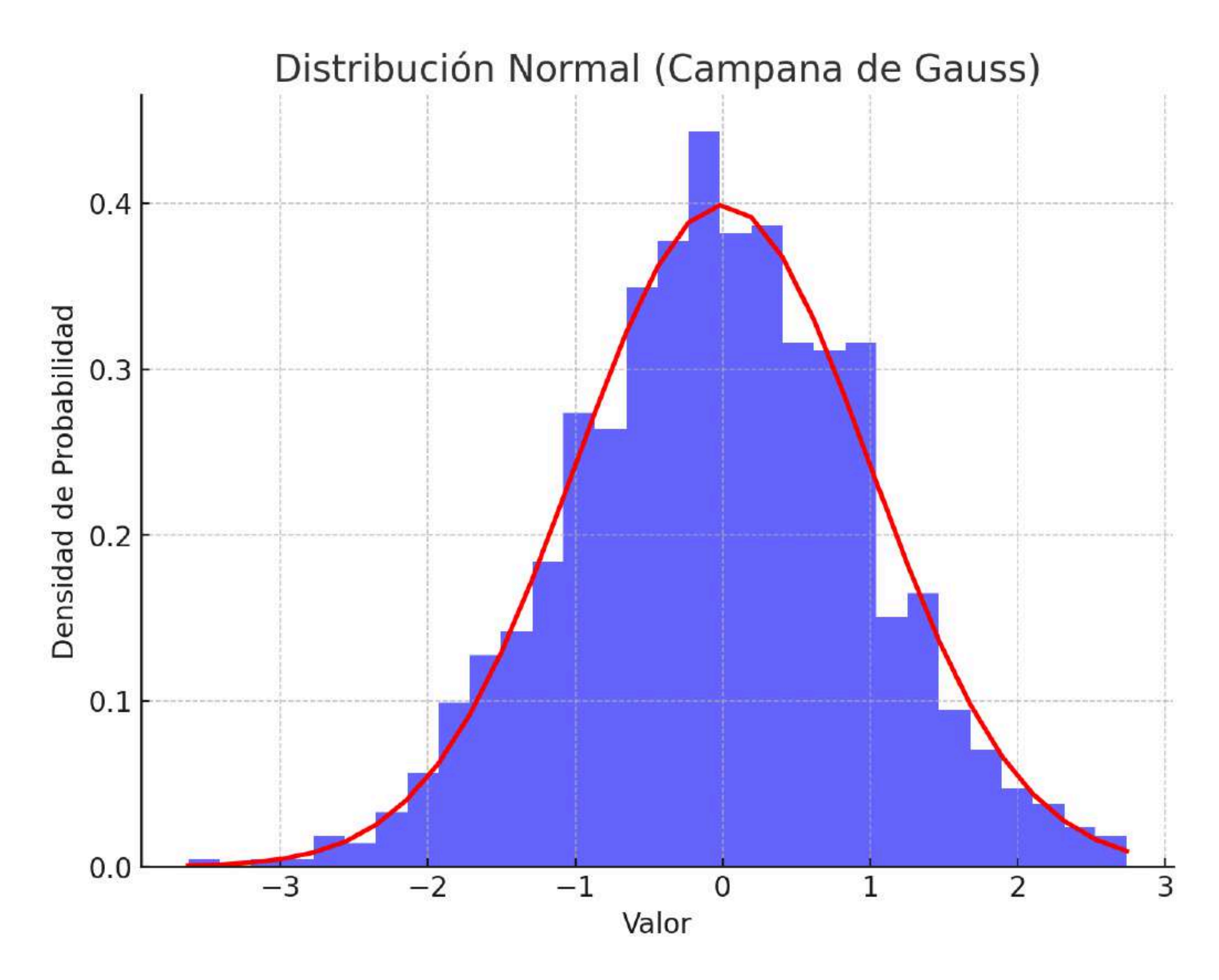


La aplicación analiza datos históricos del clima y encuentra que, en tu ciudad, la probabilidad de que llueva sigue ciertos patrones: es más alta entre las 3 y 5 de la tarde (40% de probabilidad), moderada en la mañana (20%), y muy baja en la noche (5%). Esta es una distribución de probabilidad - nos dice qué tan probable es un evento (la lluvia) en diferentes momentos del día. Con esta información, la IA puede recomendar inteligentemente horarios donde la probabilidad de lluvia sea menor, maximizando así tus posibilidades de completar tu rutina de ejercicio sin interrupciones.

Como ya te habrás dado cuenta, este concepto está totalmente relacionado con las redes bayesianas que exploramos algunas secciones atrás. Recordemos que las redes bayesianas nos permiten modelar relaciones causales entre eventos, como por ejemplo, cómo la presencia de nubes oscuras influye en la probabilidad de lluvia, o cómo la hora del día afecta la formación de nubes. Cada nodo en nuestra red bayesiana tiene asociada una distribución de probabilidad que describe qué tan probable es cada posible estado de ese nodo, dado el estado de sus nodos "padre". En nuestro ejemplo del clima, la red bayesiana no solo nos dice que es más probable que llueva en la tarde, sino que también modela cómo esta probabilidad cambia según otros factores como la humedad, la temperatura, o la velocidad del viento.



Concretamente, las distribuciones de probabilidad describen cómo se distribuyen los valores de una *variable aleatoria*[108]. Por ejemplo, la *distribución normal* (*campana de Gauss*) es común en muchas aplicaciones de IA y describe cómo se distribuyen los datos alrededor de un valor medio. Es una forma característica que aparece con frecuencia en la naturaleza y en muchos fenómenos cotidianos y se llama así porque su gráfica tiene forma de campana. Imaginá las estaturas de todas las personas adultas de un país: la mayoría tendrá una estatura cercana al promedio (el centro de la campana), mientras que habrá cada vez menos personas muy altas o muy bajas (los extremos de la campana). Este mismo patrón aparece en muchísimas situaciones, desde las calificaciones de un examen hasta medidas en biología, física y otros campos, lo que la hace especialmente útil en inteligencia artificial.



[108] Una variable aleatoria es simplemente una variable cuyo valor no podemos predecir con certeza, pero sí podemos describir qué valores puede tomar y qué tan probable es cada uno. Por ejemplo, cuando lanzamos un dado, el resultado es una variable aleatoria que puede tomar los valores 1, 2, 3, 4, 5 o 6, cada uno con igual probabilidad. Otros ejemplos de variables aleatorias son la temperatura máxima de mañana, el número de clientes que visitarán una tienda en un día, o la estatura de una persona elegida al azar.

Las ***distribuciones de probabilidad*** son importantes porque nos permiten entender y modelar la incertidumbre en los datos[109]. Al describir cómo se distribuyen los posibles valores de una *variable aleatoria*, podemos hacer predicciones, tomar decisiones informadas y calcular la probabilidad de que ocurran ciertos eventos[110]. En aplicaciones de inteligencia artificial y análisis de datos, las distribuciones nos ayudan a identificar patrones, evaluar riesgos y ajustar modelos predictivos de manera más precisa.

Distribuciones de Probabilidad en IA

Distribución Normal

La campana de Gauss es omnipresente en IA. Describe la distribución de datos alrededor de un valor medio.

Se utiliza en algoritmos de clasificación y regresión. Es fundamental en el aprendizaje profundo para inicializar pesos de redes neuronales.

Distribución Binomial

Modela eventos con dos posibles resultados. Es crucial en algoritmos de clasificación binaria.

Se aplica en modelos de regresión logística y árboles de decisión. Ayuda a evaluar la precisión de modelos predictivos.

Distribución de Poisson

Útil para modelar eventos raros en intervalos fijos. Se aplica en detección de anomalías y sistemas de recomendación.

Es esencial en modelos de predicción de tráfico y análisis de fallos en sistemas complejos.

Veamos ahora un ejemplo concreto de cómo podríamos aplicar estos conceptos en la práctica.

Control de inventarios

Un ejemplo sencillo de cómo usar distribuciones de probabilidad para tomar una decisión informada es en el

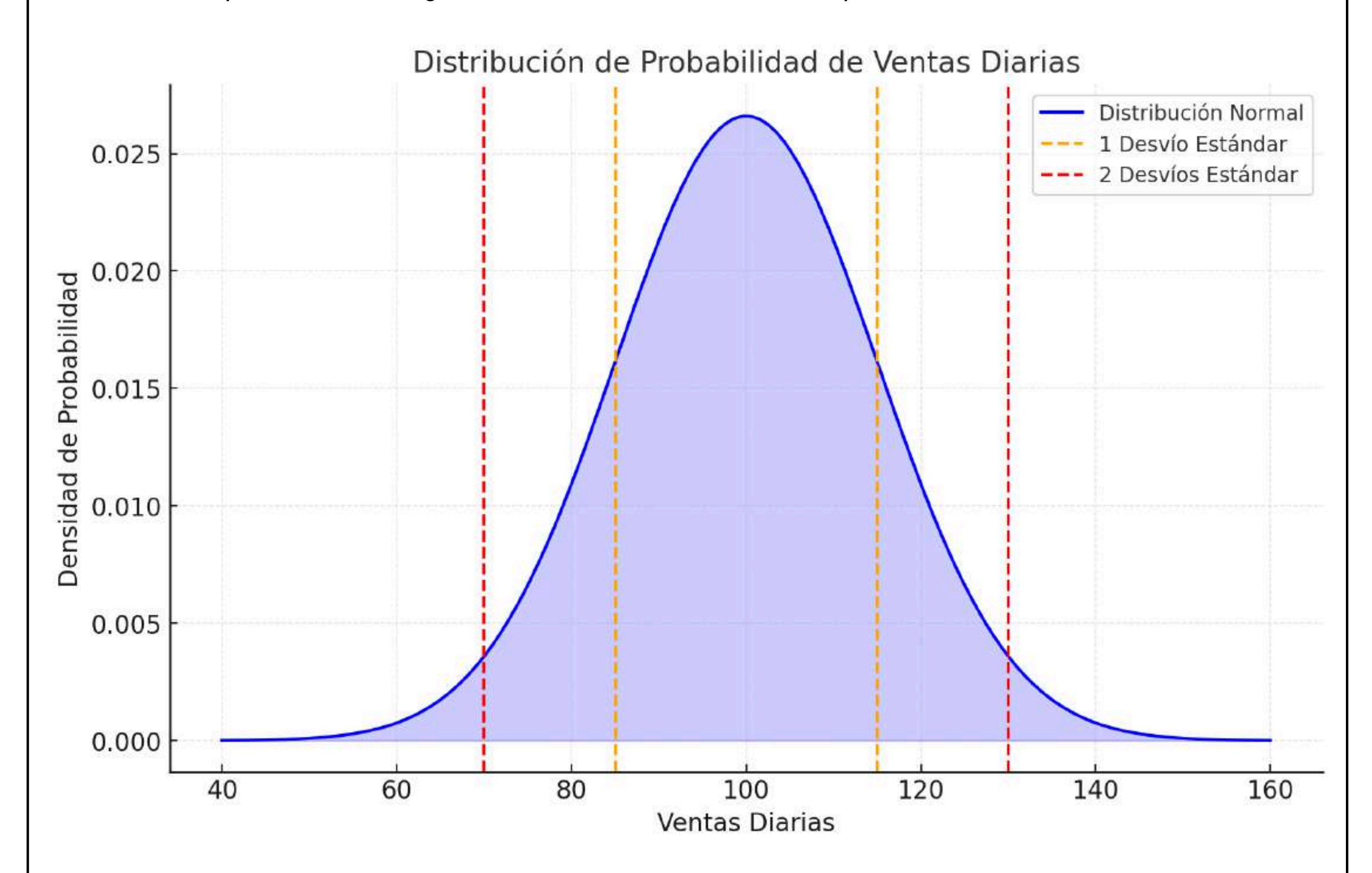
[109] Aunque la distribución normal (campana de Gauss) es muy común, existen muchas otras distribuciones de probabilidad que describen diferentes tipos de fenómenos. Por ejemplo: la distribución uniforme, donde todos los resultados son igualmente probables, como en el lanzamiento de un dado; la distribución de Poisson, que describe eventos raros como el número de clientes que llegan a un banco por hora o el número de errores en una página web; la distribución exponencial, que modela el tiempo de espera entre eventos, como el tiempo entre dos terremotos o entre dos mensajes recibidos en un servidor; y la distribución binomial, que describe situaciones de éxito/fracaso repetidas, como el número de caras que obtenemos al lanzar una moneda varias veces. Cada una de estas distribuciones tiene su propia "forma" característica y es útil para modelar diferentes situaciones del mundo real.

[110] En el caso de las estaturas, podríamos predecir qué porcentaje de la población necesitará ropa de tallas especiales, o en el caso de calificaciones de exámenes, podríamos detectar si un estudiante tiene un rendimiento inusualmente bajo o alto que requiera atención especial.

caso de **control de inventarios**.

Imaginá que sos responsable de gestionar el inventario de un producto y sabés que las ventas diarias de ese producto siguen una **distribución normal** con una *media* diaria de 100 unidades y una *desviación estándar* de 15 unidades. La media representa el valor central o "típico" - en este caso, si sumáramos las ventas de muchos días y las dividiéramos por el número de días, obtendríamos aproximadamente 100. Por su parte, la desviación estándar nos indica qué tan dispersos están los valores alrededor de esa media - una desviación estándar de 15 unidades significa que es común ver días con ventas entre 85 y 115 unidades (un desvío de la media), mientras que días con ventas por debajo de 70 o por encima de 130 unidades (dos desvíos) son mucho menos frecuentes.

A continuación podemos ver un gráfico con nuestra distribución de probabilidades:



Si tu objetivo es asegurarte de que siempre tengas suficiente inventario para satisfacer la demanda el 95% del tiempo (es decir, en el 95% de los días), podés usar la *distribución de probabilidad* para calcular cuántas unidades deberías tener siempre en stock. Sabemos que en una distribución normal, para cubrir el 95% de los casos, necesitamos considerar un margen de 1.645 veces la desviación estándar por encima de la media. Este valor 1.645 es una propiedad matemática de la distribución normal que los estadísticos han

calculado y que nos permite hacer este tipo de predicciones[111].

Por lo tanto, para calcular el stock necesario, sumamos a la media (μ) ese margen de seguridad (1.645 × σ):

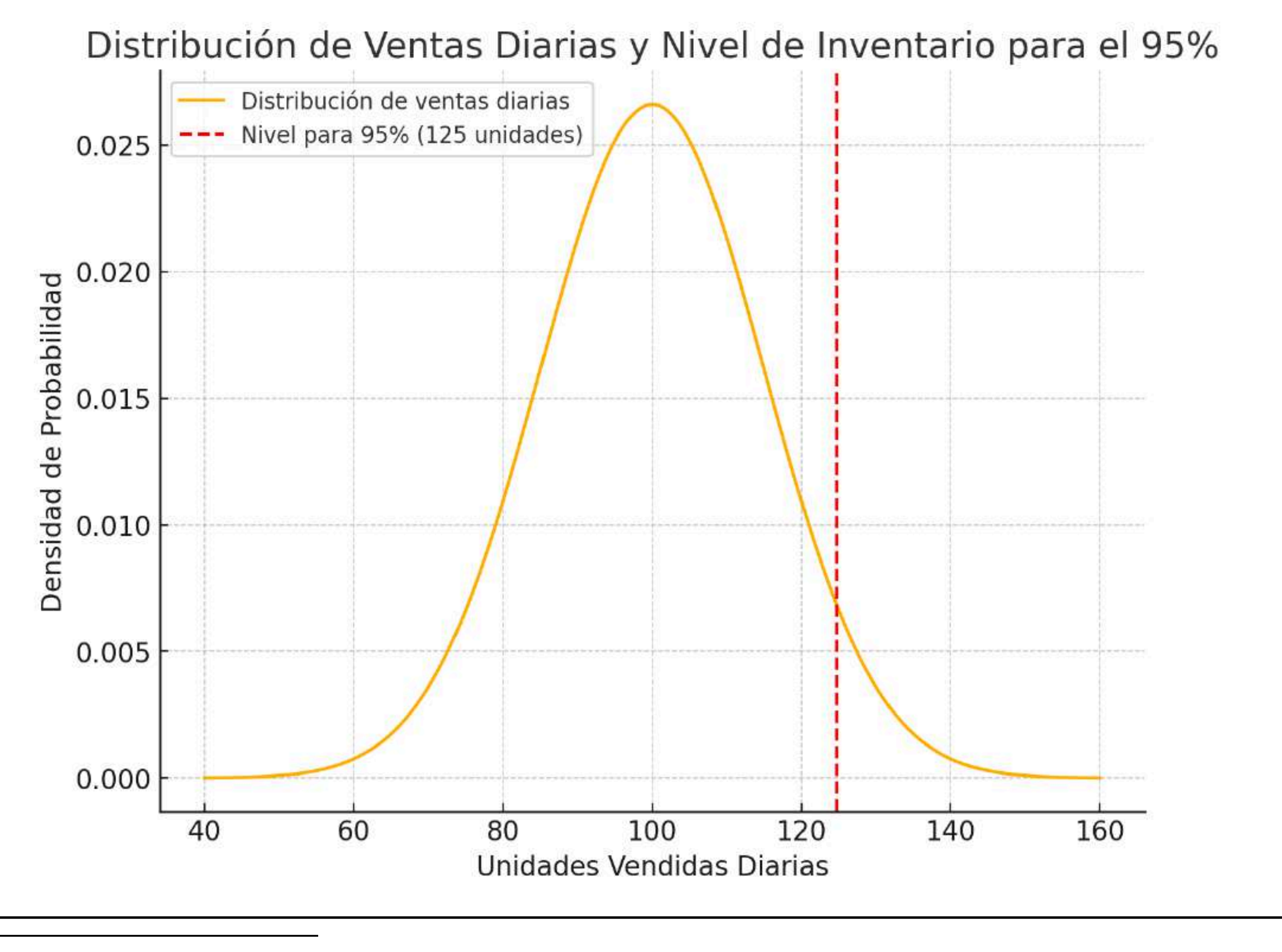
Stock necesario = media(μ) + 1.645 × desvio_estandar(σ)

= 100 + 1.645 × 15

= 124.675 ≈ 125 unidades

En este caso, podrías decidir mantener al menos 125 unidades en inventario para estar cubierto el 95% de los días. **Así, la decisión de cuánto inventario tener está informada por la distribución de probabilidad**.

No obstante, en la práctica querrás tener más precisión y ser más flexible. Un modelo de IA puede optimizar la gestión de inventarios prediciendo con mayor precisión la demanda diaria y ajustando los niveles de stock en función de factores como estacionalidad, promociones y eventos externos.



[111] En una distribución normal, existen varios valores de referencia comúnmente utilizados como margen de seguridad, dependiendo de qué tan seguros queremos estar. Por ejemplo: si usamos 1.0 desviaciones estándar, cubrimos aproximadamente el 68% de los casos; con 1.645 desviaciones estándar (como en nuestro ejemplo) cubrimos el 95% de los casos; con 2 desviaciones estándar cubrimos el 97.5% de los casos; y con 3 desviaciones estándar cubrimos el 99.7% de los casos. Es como un termómetro de seguridad: mientras más alto sea el margen que elegimos, más casos cubrimos, pero también necesitaremos mantener más inventario. Por ejemplo, si en nuestro caso quisiéramos cubrir el 99.7% de los días (usando 3 desviaciones estándar), necesitaríamos tener 145 unidades en stock en lugar de 125.

Sin embargo, la probabilidad sigue siendo crucial porque modela la incertidumbre inherente a cualquier predicción. Entender las distribuciones de probabilidad permite cuantificar el riesgo de quedarse sin stock o de tener exceso de inventario, lo que ayuda a tomar decisiones informadas.

Así, la IA y la probabilidad trabajan juntas para maximizar eficiencia y minimizar costos.

En el año 2023 tuve la oportunidad de colaborar con una empresa especializada íntegramente en la aplicación de la inteligencia artificial para la detección de actividad financiera fraudulenta. Decirlo así no es del todo cierto, realmente lo que esta empresa ofrece es un poderoso sistema de detección de todo lo que *no sea normal*. Su aplicación concreta al fraude financiero y el blanqueo de capitales[112] sean tal vez los casos más emblemáticos de la *detección de anomalías*, pero esto realmente puede ser aplicado en todo tipo de ámbitos, y es así como me tocó participar de un caso de uso que llevaba estos mismos principios a la detección de actividad sospechosa de estar relacionada con el *tráfico de personas*.

Si, lo que te estoy contando.

Esta actividad inhumana en la práctica debe ser financiada y debe de alguna forma u otra movilizar recursos dentro del sistema interno e internacional de pagos, intentando hacer maniobras con el fin de pasar desapercibida por los controles internos de los bancos. Estos sistemas, hasta hace no mucho tiempo eran práctica o totalmente basados en reglas (como los sistemas expertos que vimos anteriormente). Reglas que los criminales con el tiempo pueden aprender y jugar con ellas hasta el límite. Afortunadamente esto ya está cambiando y la inteligencia artificial está jugando un rol fundamental en ponerle freno a esas actividades.

Me encanta entonces poder dar este ejemplo por que para el caso de tráfico de personas que te comenté, afortunadamente me tocó vivir casos concretos de éxito donde la combinación de probabilidad y estadística bien aplicada, sumada a unos algoritmos extraordinarios de *machine learning* no supervisado permitió detectar actividad sospechosa y confirmar casos que pasaron a ser investigados y reportados. La estadística y la IA al servicio de la humanidad como debería ser siempre.

La estadística y la probabilidad son la base para que estos algoritmos hagan su trabajo y es por eso que es tan importante que tengas un buen entendimiento de los conceptos básicos.

[112] El fraude financiero se refiere a cualquier acto intencional que busca obtener una ganancia financiera de manera engañosa o deshonesta, como por ejemplo el uso no autorizado de tarjetas de crédito o la falsificación de documentos bancarios. Por su parte, el blanqueo de capitales es el proceso mediante el cual se intenta dar apariencia legal a fondos que provienen de actividades ilícitas, generalmente a través de una serie de transacciones diseñadas para ocultar el origen real del dinero. Las instituciones financieras están obligadas por ley a implementar sistemas para detectar y prevenir estas actividades, y es acá donde las técnicas de análisis estadístico y la IA juegan un papel fundamental, al poder identificar patrones de transacciones inusuales o sospechosas que se desvían significativamente del comportamiento financiero normal de los clientes.

Veamos otro ejemplo.

El uso de la probabilidad y estadística en la detección de fraude

La detección de fraude en transacciones financieras es una aplicación crítica de la probabilidad y la estadística en la inteligencia artificial. Utilizamos estos campos para identificar patrones inusuales en las transacciones que podrían indicar *fraude*. Por ejemplo, cuando un sistema de IA analiza miles de transacciones bancarias, aprende el comportamiento típico de cada cliente: sus patrones de gasto habituales, las ubicaciones desde donde suele operar, los horarios comunes de sus transacciones y los montos que normalmente maneja. Cuando aparece una transacción que se desvía significativamente de estos patrones - como una compra en un país donde el cliente nunca operó, por un monto inusualmente alto, o en un horario atípico - el sistema calcula la probabilidad de que esta transacción sea legítima basándose en el historial del cliente. Si esta probabilidad es muy baja, el sistema puede marcar la transacción como potencialmente fraudulenta para su revisión.

Veamos cómo podríamos hacerlo paso por paso.

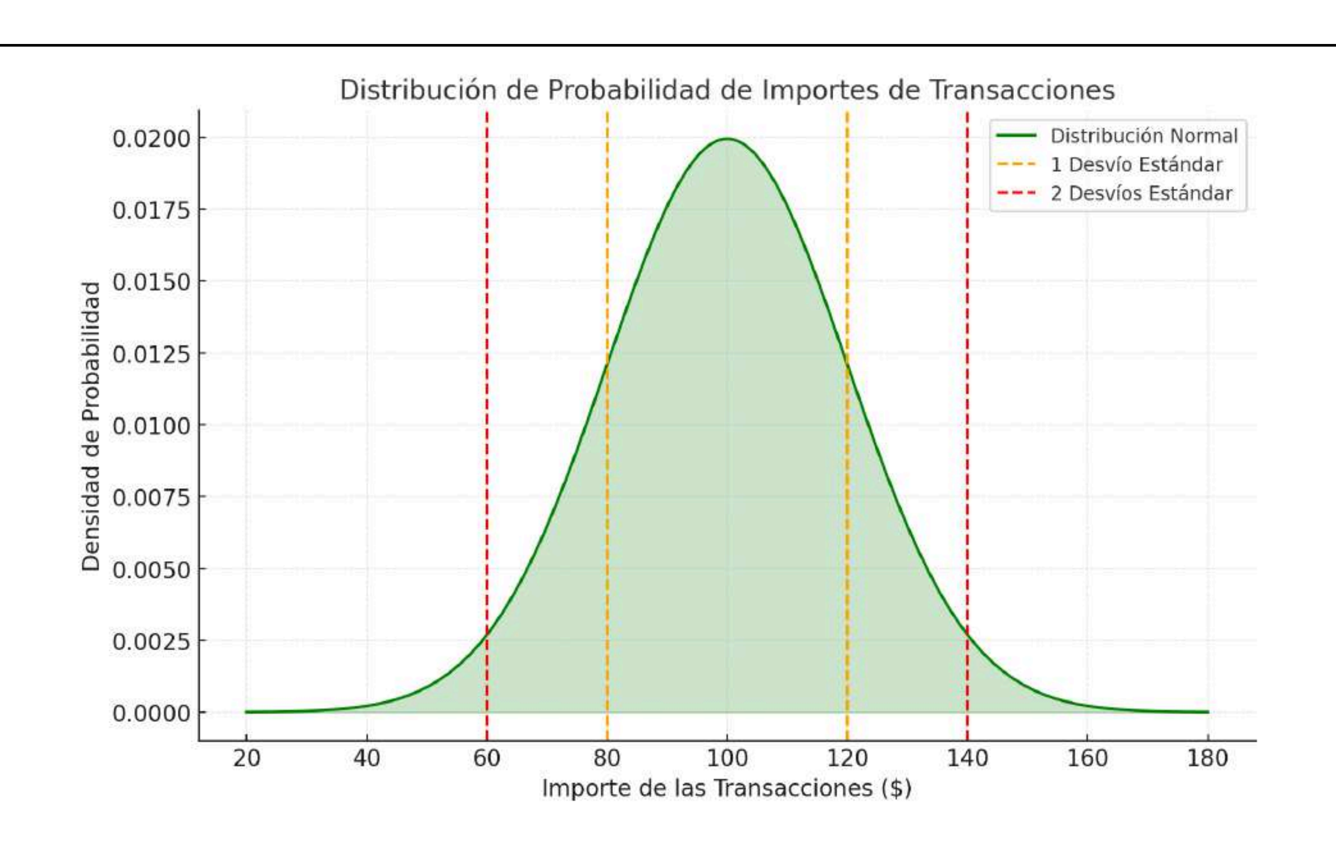
Paso 1: Recopilación de Datos

Primero, recopilamos datos sobre las transacciones financieras de la entidad con la que estamos trabajando. Estos datos pueden incluir información como el monto de la transacción, la ubicación, la hora del día, el método de pago y el historial de transacciones del usuario. Esta información se organiza en una tabla donde cada fila representa una transacción y cada columna representa una característica de esa transacción. Los datos se trabajan y se preparan de una forma en la que puedan ser analizados los perfiles de los usuarios. Es decir, estos sistemas tratan de identificar y estudiar a individuos y no a una transacción aislada.

Paso 2: Análisis de Distribuciones de Probabilidad

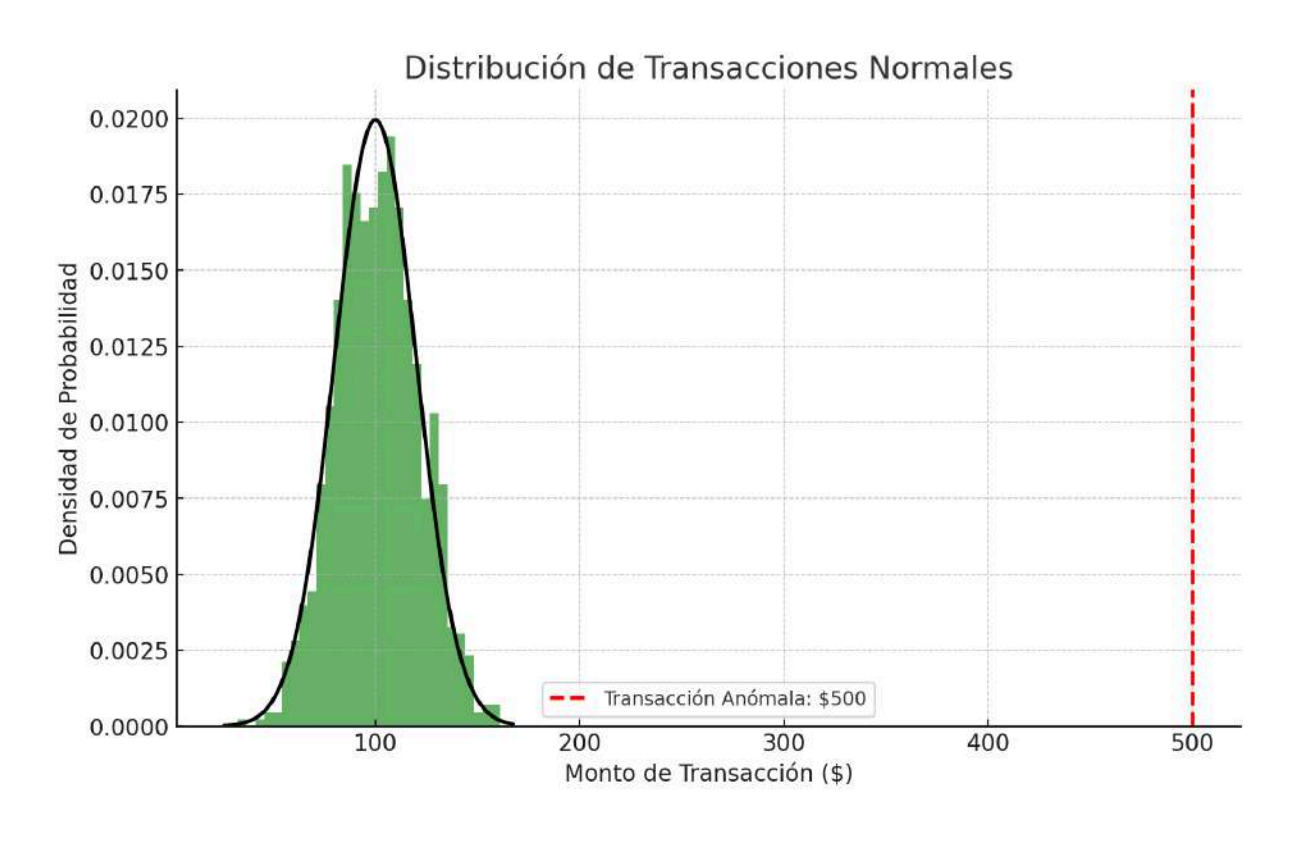
Utilizamos distribuciones de probabilidad para modelar las características normales de las transacciones. Por ejemplo, podemos analizar la distribución de los importes de las transacciones. Supongamos que descubrimos que la mayoría de las transacciones de un usuario siguen una distribución normal con una media de 100 dólares y una desviación estándar de 20 dólares.[113]

[113] En la práctica lo usual es no solamente mirar los montos de las transacciones sino otra serie de características relacionadas a la transacción. Las llamamos *features* y de cada una de de ellas se analiza su distribución de probabilidad con el objetivo nuevamente de saber qué es lo *normal* y de esa forma, qué sería *anómalo* bajo esas condiciones.



Paso 3: Cálculo de Probabilidades

Calculamos la probabilidad de que una nueva transacción sea legítima basándonos en las distribuciones de probabilidad que hemos modelado. Si una nueva transacción de 500 dólares ocurre, podemos calcular la probabilidad de que esta transacción sea *normal* según nuestra distribución.



Paso 4: Identificación de Transacciones Anómalas

Si la probabilidad de que una transacción sea normal es muy baja (por ejemplo, menos del 1%), etiquetamos esta transacción como potencialmente fraudulenta. Utilizamos técnicas estadísticas como el *test de hipótesis*[114] para determinar si una transacción se desvía significativamente de la norma.

Paso 5: Sistema de Alerta

Desarrollamos un sistema de alerta que marque las transacciones con baja probabilidad como sospechosas. Estas transacciones pueden ser revisadas manualmente por un analista o por un sistema automatizado más avanzado para determinar si realmente son fraudulentas.

Ejemplo utilizando *z-scores*

Supongamos que un usuario normalmente realiza transacciones de entre $80 y $120. Analizamos los datos históricos y encontramos que las transacciones siguen una distribución normal con una media de $100 y una desviación estándar de $20 (iguales gráficos que los anteriores).

Distribución Normal de Transacciones:

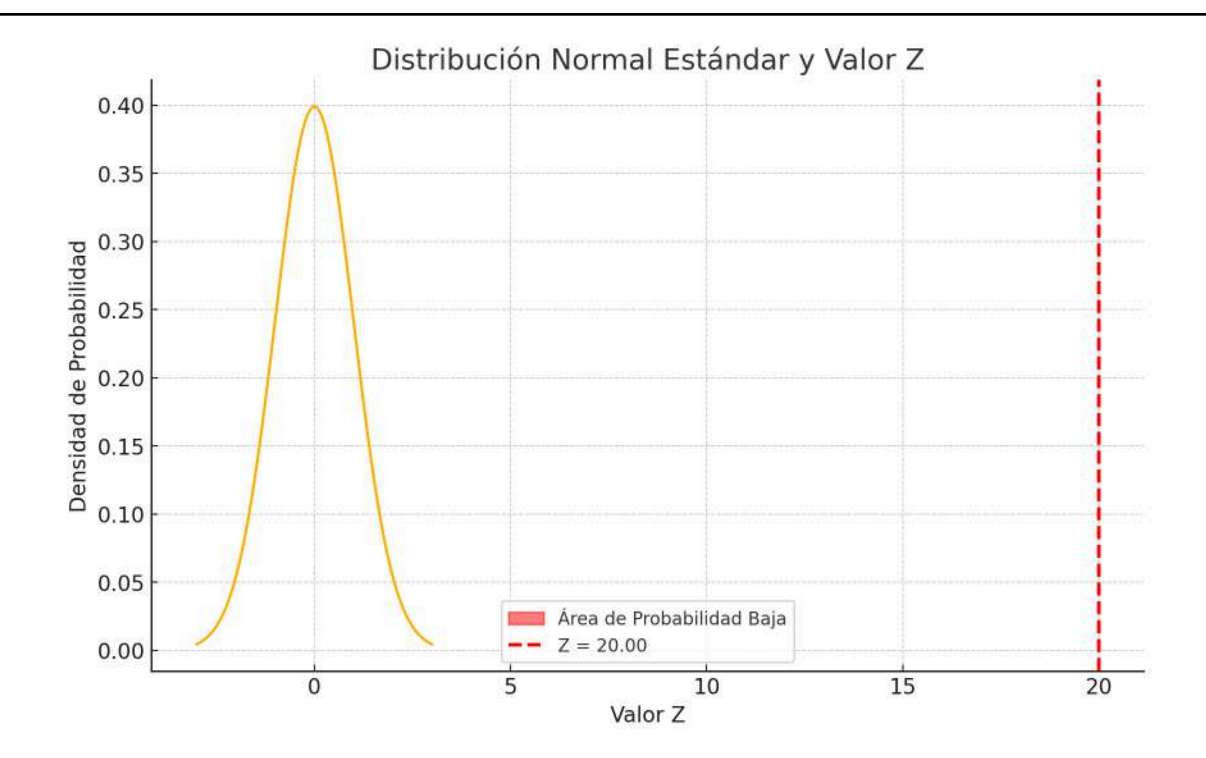
- Media (mu): $100
- Desviación estándar (sigma): $20

Ahora, una nueva transacción de $500 se realiza. Queremos calcular la probabilidad de que esta transacción sea normal.

ID	Monto ($)	Ubicación	Hora	Método de Pago
1	95	Nueva York	14:30	Tarjeta de Crédito
2	110	Los Ángeles	09:45	PayPal
3	500	Miami	02:15	Tarjeta de Débito

Para esto podemos usar un **z-score**. El z-score es una herramienta estadística fundamental en este caso porque nos permite cuantificar **qué tan inusual es** una transacción en relación con el comportamiento típico del usuario.

[114] Un test de hipótesis es una herramienta estadística que nos ayuda a tomar decisiones basadas en datos, evaluando si una observación puede explicarse por el azar o si representa una diferencia significativa que requiere atención. En el contexto de detección de fraude, el test plantea dos hipótesis: la hipótesis nula (H_0) que asume que la transacción es normal, y la hipótesis alternativa (H_1) que sugiere que la transacción es inusual. El test calcula la probabilidad (llamada "valor p") de observar una transacción tan extrema como la analizada si fuera realmente normal. Si esta probabilidad es muy baja (típicamente menor al 1% o 5%), "rechazamos la hipótesis nula" - es decir, concluimos que la transacción es suficientemente inusual como para justificar una investigación más detallada. Esta metodología nos permite tomar decisiones objetivas basadas en evidencia estadística en lugar de intuiciones o reglas arbitrarias.



Específicamente, el z-score nos indica **cuántas desviaciones estándar se aleja un valor de la media**, traduciendo así el monto de la transacción a una medida estandarizada que podemos interpretar fácilmente, independientemente de la escala monetaria. Se calcula así:

- **Calcular el valor Z (z-score)**:
  - Z = (valor - mu) / sigma
  - Z = (500 - 100) / 20
  - Z = 20
- **Buscar el valor Z en una tabla de distribución normal**: En el caso de una transacción de $500 que resulta en un z-score de 20, esto significa que el valor está a 20 desviaciones estándar por encima de la media, lo cual es extremadamente raro en una distribución normal (donde aproximadamente el 99.7% de los valores caen dentro de ±3 desviaciones estándar). Esta medida nos permite establecer umbrales objetivos y automatizables para identificar transacciones atípicas que merecen una revisión más detallada.
- **Conclusión**: Dado que la probabilidad de que una transacción de $500 sea normal es extremadamente baja, marcamos esta transacción como potencialmente fraudulenta y activamos una alerta para su revisión.

> **¡Alerta de Fraude!** La transacción de $500 ha sido marcada como potencialmente fraudulenta y requiere revisión.

Un modelo de IA avanzado podría mejorar significativamente este proceso de detección de fraudes al incorporar múltiples factores además del monto de la transacción. Por ejemplo, podría analizar patrones de comportamiento del usuario, como la ubicación geográfica, la hora del día, el tipo de comercio, y la frecuencia de transacciones. Utilizando algoritmos de *machine learning* el modelo podría identificar patrones

complejos y correlaciones no lineales en los datos que serían difíciles de detectar con métodos estadísticos tradicionales. Además, un sistema de IA podría adaptarse y aprender continuamente de nuevos datos, mejorando su precisión con el tiempo y ajustándose rápidamente a nuevas tácticas de fraude. Esto no sólo reduciría los falsos positivos, minimizando las molestias para los usuarios que no están cometiendo ningún crimen, sino que también podría detectar fraudes más sutiles que podrían pasar desapercibidos con métodos más simples.

Este ejemplo de detección de fraude ilustra perfectamente lo que mencionamos al inicio del capítulo: cómo las matemáticas y la estadística son el lenguaje fundamental que permite a la IA resolver problemas del mundo real. Al modelar el comportamiento normal de las transacciones mediante distribuciones de probabilidad, estamos traduciendo un problema complejo a un lenguaje que la IA puede entender y procesar. No necesitás ser un experto en estadística para apreciar la lógica detrás: así como un cajero bancario aprende a reconocer patrones sospechosos basándose en su experiencia, la IA utiliza matemáticas para cuantificar y automatizar este proceso de detección a una escala mucho mayor.

Con estos fundamentos matemáticos y estadísticos que exploramos, desde el álgebra y el cálculo hasta las distribuciones de probabilidad, estás mejor equipada/o para entender cómo "piensa" la IA y cómo puede aplicarse a diferentes contextos.

Ahora sí, con esta base sólida, podemos adentrarnos en casos de uso emblemáticos de la IA que están transformando industrias enteras. Seguime.

Casos de uso de la Inteligencia Artificial

Llegados a esta sección (y aún en el *Segundo Paso*[115]) podemos afirmar que ya conocés los básicos de la inteligencia artificial y además tenés una mejor noción de cómo las matemáticas y la estadística proporcionan el lenguaje en el cual *hablan* cada uno de ellos[116]. Ahora es momento de concretar el paso y profundizar en algunos **casos de uso** de la IA con los que nos encontramos frecuentemente tanto en la vida cotidiana como en el mundo empresarial, incluyendo tecnologías de IA generativa con modelos de lenguaje (LLMs).

01. 🎯 Hiper-personalización de ofertas y contenido



Este es uno de los ejemplos más reconocibles de los últimos 10-15 años, la capacidad de la inteligencia artificial de personalizar servicios y productos a cada individuo tal que sintamos como que "nos conocen". La IA analiza el comportamiento de compra, el historial y las preferencias de los clientes para ofrecer recomendaciones de productos personalizadas. Este nivel de personalización no solo mejora la experiencia del usuario, sino que también

[115] Ya lo seeee. Es un paso muy largo ¿no? jajaja

[116] No lo veremos en este libro, pero acá faltaría mencionar a los lenguajes de programación. Los lenguajes de programación son herramientas esenciales en la inteligencia artificial, ya que permiten implementar los conceptos matemáticos y estadísticos en aplicaciones prácticas. Aunque no se cubren en este material, son fundamentales para dar vida a las ideas de IA en el mundo real. Entre los lenguajes más utilizados en IA se encuentran Python, debido a su simplicidad y amplia gama de bibliotecas especializadas; R, popular en análisis estadístico y machine learning; Java y C++, preferidos para aplicaciones de alto rendimiento. La elección del lenguaje depende por lo general del tipo de proyecto de IA y de las necesidades específicas de rendimiento y escalabilidad.

aumenta la eficiencia al reducir el tiempo de búsqueda y descubrimiento de nuevos productos o contenidos. Plataformas como Netflix y Amazon utilizan algoritmos de *machine learning* para analizar nuestros hábitos de consumo y ofrecernos productos o contenidos que coincidan con nuestros gustos y preferencias anteriores o con el de otros usuarios similares a nosotros.

Hiperpersonalización: Revolucionando la Experiencia del Cliente

La inteligencia artificial está transformando la manera en que las empresas interactúan con sus clientes. La hiperpersonalización utiliza IA para analizar el comportamiento del usuario y ofrecer recomendaciones altamente personalizadas.

Este enfoque mejora la experiencia del cliente y aumenta la eficiencia en la búsqueda de productos y contenidos.



¿Cómo se genera una recomendación personalizada?

Para entender cómo funcionan las recomendaciones personalizadas, veamos un ejemplo teórico y las técnicas de IA que se utilizan. Primero, como siempre, se recopilan datos.

Recopilación de Datos

En este caso, los datos son acerca del comportamiento del usuario con una plataforma digital. Estos datos pueden incluir:

- Historial de compras
- Calificaciones que ha dado a productos

- Tiempo de visualización de contenidos
- Interacciones (clics, comentarios, etc.)
- etc.

Datos de Interacciones de Usuario con Productos

fecha_hora	id_usuario	id_producto	tipo_interaccion	valor	duracion_sesion_seg	tipo_dispositivo
2024-01-15 10:23:45	usuario_123	prod_456	visualización	1	45	móvil
2024-01-15 10:25:12	usuario_123	prod_456	calificación	4	120	móvil
2024-01-15 11:30:00	usuario_456	prod_789	compra	1	300	escritorio
2024-01-15 12:15:30	usuario_789	prod_123	visualización	1	25	tablet
2024-01-15 12:20:45	usuario_789	prod_456	clic	1	15	tablet
2024-01-15 13:45:22	usuario_123	prod_789	agregar_carrito	1	90	móvil
2024-01-15 14:10:15	usuario_456	prod_123	calificación	5	60	escritorio
2024-01-15 15:20:30	usuario_789	prod_789	compra	1	180	tablet
2024-01-15 16:05:45	usuario_123	prod_123	visualización	1	30	escritorio
2024-01-15 16:30:00	usuario_456	prod_456	agregar_carrito	1	75	móvil

Posteriormente, los datos se pueden organizar en una matriz de usuario-artículo, donde las filas representan usuarios y las columnas representan artículos. Cada celda de la matriz contiene una puntuación que representa la interacción del usuario con el artículo, como una calificación o la cantidad de veces que se ha visto un contenido.

Matriz de Interacciones Usuario-Artículo

Usuario	Producto A	Producto B	Producto C	Producto D	Producto E
Usuario_1	5★	3★	-	4★	2★
Usuario_2	-	4★	5★	3★	4★
Usuario_3	2★	-	4★	5★	3★
Usuario_4	4★	5★	3★	-	5★

Leyenda:
- 1★ - 5★: Calificación del usuario
- - : No hay interacción

Filtrado Colaborativo

Con estos datos, una opción para generar recomendaciones es usar *filtrado colaborativo.* El filtrado colaborativo es una técnica comúnmente utilizada en los sistemas de recomendación y hay dos tipos principales:

1. **Filtrado colaborativo basado en el usuario**:
    - Se identifican usuarios similares basándose en sus interacciones con los **artículos**.
    - Se recomiendan artículos que han sido bien recibidos por usuarios similares.
2. **Filtrado colaborativo basado en el artículo**:
    - Se identifican artículos similares basándose en las interacciones de los **usuarios**.
    - Se recomiendan artículos similares a los que el usuario ya ha interactuado positivamente.

Análisis de Filtrado Colaborativo

1. Matriz de Similitud entre Usuarios

Usuario	usuario_123	usuario_456	usuario_789
usuario_123	1.00	0.85	0.62
usuario_456	0.85	1.00	0.73
usuario_789	0.62	0.73	1.00

Los valores representan la similitud entre usuarios (1.00 = idéntico, 0.00 = sin similitud)

2. Recomendaciones Generadas

Usuario	Producto Recomendado	Score	Razón
usuario_123	prod_123	0.92	Basado en calificación alta de usuario_456 (similar)
usuario_456	prod_456	0.88	Basado en interacciones positivas de usuario_123
usuario_789	prod_456	0.78	Basado en compras similares de usuario_456

¿Cómo se generaron estas recomendaciones?

1. Se analizaron todos los tipos de interacciones (visualizaciones, clics, calificaciones, compras)
2. Se calculó la similitud entre usuarios basándose en patrones de comportamiento comunes
3. Para cada usuario, se identificaron los productos que no han interactuado pero que usuarios similares sí han valorado positivamente
4. Se generó un score de recomendación basado en la similitud del usuario y la fuerza de las interacciones positivas

Análisis predictivo

El análisis predictivo en sistemas de recomendación aplica algoritmos de *machine learning*, como la *descomposición en valores singulares* (SVD) y las redes neuronales, para anticipar las preferencias de los usuarios sobre artículos con los que aún no han interactuado. A diferencia del filtrado colaborativo tradicional, este enfoque no se limita solo a analizar las interacciones usuario-artículo, sino que incorpora un nivel más amplio de información, incluyendo datos demográficos, patrones de navegación, historiales de

compra y factores contextuales como tendencias de mercado o estacionalidad, permitiendo así predicciones más precisas y personalizadas.

fecha_hora	id_usuario	id_producto	categoria_producto	precio	interaccion	valor_interaccion	edad_usuario	genero	ubicacion	dispositivo	hora_del_dia	dia_semana	temp_ambiente	ultima_compra_dias	descuento_aplicado
15/01/2024 9:30	user_123	prod_789	ropa_deportiva	89.99	compra	1	28	F	Buenos Aires	móvil	9	1	12	15	0
15/01/2024 10:15	user_456	prod_234	tecnologia	299.99	visualización	1	35	M	Cordoba	escritorio	10	1	14	7	0.15
15/01/2024 11:45	user_789	prod_567	hogar	45.50	calificación	4	42	F	Mendoza	tablet	11	1	16	3	0.1
15/01/2024 13:20	user_123	prod_890	ropa_deportiva	65.00	agregar_carrito	1	28	F	Buenos Aires	móvil	13	1	18	0	0.2
15/01/2024 14:45	user_456	prod_123	tecnologia	199.99	compra	1	35	M	Tucumán	escritorio	14	1	19	0	0.25

Los algoritmos de machine learning empleados en el análisis predictivo, como las redes neuronales, no solo identifican patrones en los datos existentes, sino que también aprenden a generalizar estos patrones para hacer predicciones sobre nuevas situaciones. Por ejemplo, podrían predecir qué productos es más probable que un usuario compre en el futuro, incluso si son productos que nunca ha comprado antes. Esto se logra analizando múltiples variables y sus interrelaciones, permitiendo al sistema hacer predicciones más sofisticadas y personalizadas que van más allá de simples correlaciones entre usuarios o artículos.

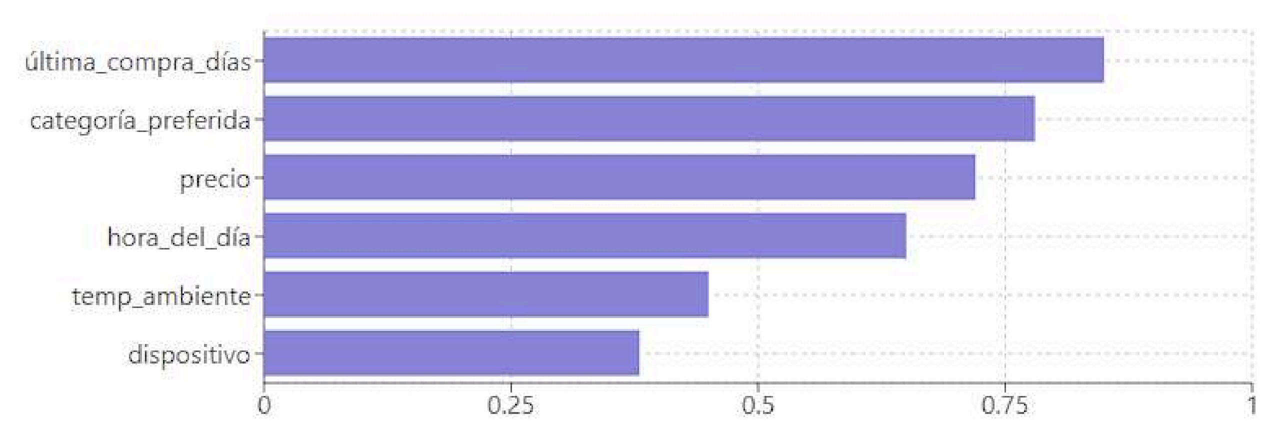


Precisión del Modelo
87%

Recall
83%

F1-Score
85%

Importancia de Características



La ilustración anterior nos muestra dos aspectos clave del análisis predictivo del comportamiento de usuarios. En la parte superior, vemos tres métricas fundamentales: la *precisión* del modelo (87%), que indica

qué tan acertadas son las predicciones; el *recall* (83%), que muestra la capacidad de identificar casos positivos; y el *F1-Score* (85%), que representa el balance entre las dos métricas anteriores.

En la parte inferior, el gráfico de barras muestra la importancia relativa de diferentes *características* en las predicciones, donde el tiempo desde la última compra (0.85) y la categoría preferida (0.78) son los factores más influyentes, mientras que la temperatura ambiente (0.45) y el dispositivo usado (0.38) tienen menor impacto. Esta jerarquía ayuda a entender qué factores son más relevantes al momento de predecir el comportamiento de compra de los usuarios.

Predicciones Personalizadas

Usuario	Producto	Prob. Compra	Razón Principal	Mejor Momento	Precio Óptimo
user_123	prod_567	0.89	Alta afinidad con categoría + Patrón temporal favorable	Entre 18:00-20:00	75.99€
user_456	prod_890	0.75	Historial de compras similares + Descuento aplicable	Entre 12:00-14:00	149.99€
user_789	prod_234	0.62	Navegación reciente + Temperatura favorable	Entre 09:00-11:00	89.99€

Esta tabla muestra predicciones personalizadas para tres usuarios diferentes, indicando qué tan probable es que compren productos específicos y cuál sería el mejor momento y precio para ofrecérselos. Por ejemplo, el user_123 tiene una alta probabilidad de compra (89%) del producto prod_567 debido a su afinidad con la categoría y un patrón temporal favorable, siendo el mejor momento entre las 18:00-20:00 con un precio óptimo de 75.99€; mientras que el user_456 tiene una probabilidad del 75% de comprar el prod_890 basado en compras similares previas y un descuento aplicable, con un mejor momento entre las 12:00-14:00 y un precio óptimo de 149.99€; finalmente, el user_789 muestra una probabilidad más moderada (62%) de comprar el prod_234, basándose en su navegación reciente y la temperatura favorable, siendo la mejor hora entre las 09:00-11:00 con un precio óptimo de 89.99€.

El **análisis predictivo** y el **filtrado colaborativo** se complementan eficazmente para crear un sistema de recomendación robusto: mientras el análisis predictivo permite personalizar la experiencia de compra identificando el momento óptimo, precio adecuado y probabilidad de compra basándose en patrones individuales y contextuales, el filtrado colaborativo fortalece estas predicciones al identificar similitudes entre usuarios y aprovechar las preferencias colectivas, como vimos en las matrices de similitud donde usuarios con comportamientos parecidos (por ejemplo, con índices de similitud superiores a 0.80) tienden a interesarse por productos similares. La combinación de ambos enfoques resulta en recomendaciones más precisas y relevantes, mejorando tanto la satisfacción del usuario como las tasas de conversión.

Imaginemos que sos un usuario de ***Netflix***. El proceso de recomendación mediante análisis predictivo podría ser así:

1. **Recopilación de datos**: Netflix analiza tu historial de visualización, incluyendo qué películas viste, cuáles calificaste y cuánto tiempo pasás viendo ciertos géneros.

Datos de Interacciones de Usuario en Netflix

fecha_hora	id_usuario	titulo	genero	director	tipo_interaccion	tiempo_vista_min	completado	calificacion	dispositivo
2024-01-16 15:45:00	user_123	Interestelar	Ciencia Ficción	Christopher Nolan	visualización	169	Sí	5	Laptop
2024-01-17 21:00:00	user_123	Matrix	Ciencia Ficción	Wachowski	visualización	136	Sí	4	Smart TV
2024-01-18 19:15:00	user_123	El Show de Truman	Drama	Peter Weir	visualización	103	Sí	4	Móvil
2024-01-19 22:30:00	user_123	Ex Machina	Ciencia Ficción	Alex Garland	visualización	108	Sí	4	Smart TV
2024-01-20 16:00:00	user_123	La Llegada	Ciencia Ficción	Denis Villeneuve	visualización	116	Sí	5	Laptop
2024-01-21 14:20:00	user_123	Memento	Suspense	Christopher Nolan	visualización	113	Sí	5	Tablet
2024-01-22 20:45:00	user_123	Blade Runner 2049	Ciencia Ficción	Denis Villeneuve	abandono	45	No	-	Smart TV
2024-01-23 18:30:00	user_123	El Código Enigma	Drama	Morten Tyldum	visualización	114	Sí	3	Laptop
2024-01-24 21:15:00	user_123	Primer	Ciencia Ficción	Shane Carruth	visualización	77	Sí	4	Smart TV

2. **Análisis de patrones**: El sistema nota que viste muchas películas de ciencia ficción, especialmente aquellas con elementos de viajes en el tiempo, y que tendés a ver películas completas de directores como Christopher Nolan.
3. **Identificación de características**: Basándose en este análisis, el sistema identifica que te gustan las tramas complejas, los efectos visuales impresionantes y las narrativas no lineales.

Características Preferidas Identificadas

Característica	Peso en Predicción	Ejemplo en "Origen"
Tramas complejas	90%	Sueños dentro de sueños
Efectos visuales	85%	Efectos de gravedad cero
Narrativa no lineal	80%	Múltiples líneas temporales
Director reconocido	75%	Christopher Nolan
Actores famosos	60%	Leonardo DiCaprio

4. **Aplicación de algoritmos de ML**: Netflix utiliza estos datos para alimentar sus algoritmos de *machine learning*, que predicen qué otras películas podrían gustarte.
5. **Predicción de puntuaciones**: El sistema predice cómo calificarías películas que aún no viste, basándose en sus características y en cómo se alinean con tus preferencias.
6. **Recomendación**: Finalmente, Netflix podría recomendarte una película como "Origen" (si aún no la viste). Aunque no es una película nueva, el sistema predice que te gustará porque:
   - Es de ciencia ficción
   - Tiene una trama compleja y no lineal
   - Está dirigida por Christopher Nolan
   - Tiene efectos visuales impresionantes



Este proceso de análisis predictivo permite a Netflix hacer recomendaciones personalizadas, incluso para películas con las que no interactuaste directamente, basándose en un análisis profundo de tus preferencias y comportamientos de visualización.



Modelos de Lenguaje y Generación de Contenido

Los modelos de lenguaje grande (LLMs) como ChatGPT están transformando nuestra interacción con la tecnología. Pueden generar texto coherente y relevante en diversos contextos.

Lo primero que quiero remarcar es que a pesar de que esta sección la haré mencionando puntualmente a **ChatGPT**, no pienso que sea ni la única ni la mejor alternativa. A mi personalmente me encanta **Claude** de **Anthropic** y soy muy pro open-source[117], por lo tanto también me encanta **Llama** de **Meta** y los modelos de **Mistral AI**. Dicho esto, creo que para el público general resonará más esta sección utilizando ChatGPT, así que vamos ;)

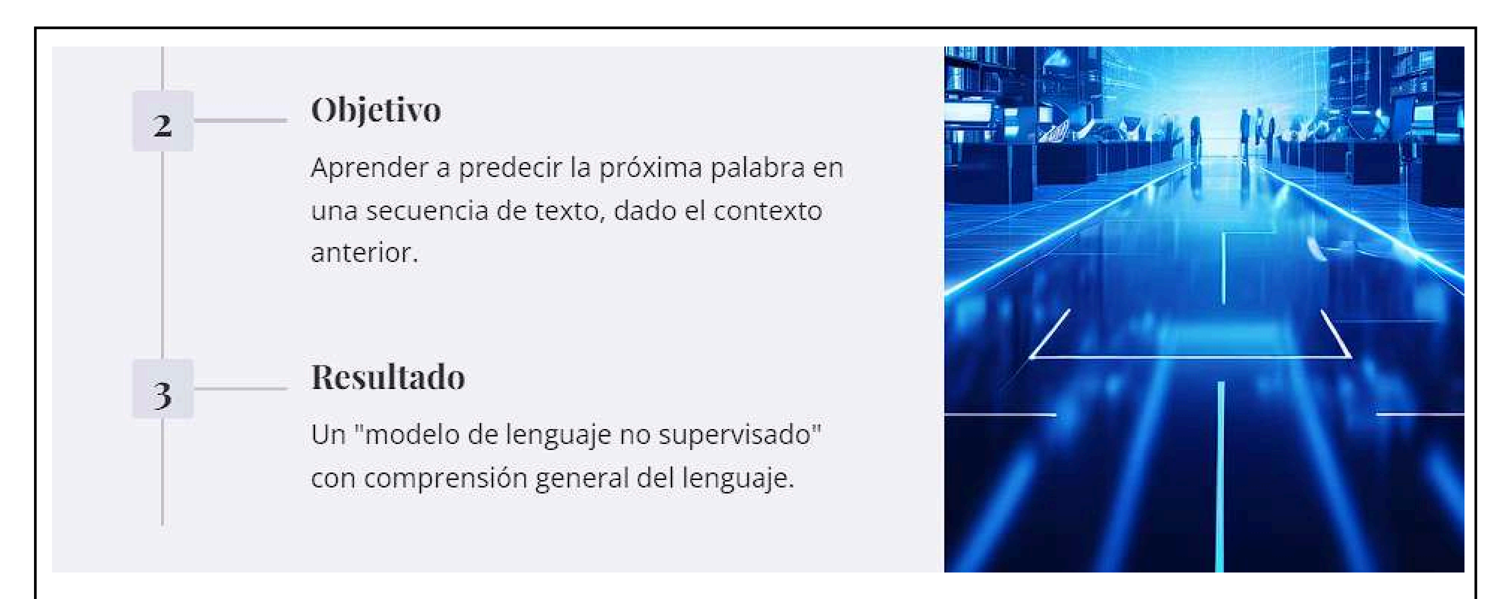
[117] De hecho, a fecha de mi última revisión del libro, el 2 de febrero de 2025, se lanzó hace pocos sías DeepSeek R1, un modelo de inteligencia artificial de código abierto desarrollado por la empresa china DeepSeek. Este modelo viene generando un impacto significativo en la industria tecnológica debido a su eficiencia y rendimiento comparables a los principales modelos estadounidenses, pero con un costo de desarrollo (presuntamente) mucho menor. Además, su naturaleza de código abierto facilita el acceso y la innovación en el campo de la inteligencia artificial.

Vamos a profundizar en cómo funciona ChatGPT, desde su entrenamiento inicial hasta su *ajuste fino* y el uso de aprendizaje por refuerzo (Reinforcement Learning).

¿Cómo Funciona ChatGPT?

Para entender cómo ChatGPT puede generar texto coherente, es importante conocer los pasos clave en su desarrollo y funcionamiento:

1. Entrenamiento Inicial del Modelo



El entrenamiento de ChatGPT comienza con un proceso llamado *preentrenamiento* utilizando una gran cantidad de texto disponible en internet. Este paso es crucial para que el modelo aprenda las estructuras del lenguaje, el significado de las palabras y las relaciones entre las frases.

- **Datos**: El modelo se entrena con un conjunto de datos masivo que incluye libros, artículos, sitios web, y más.
- **Objetivo**: Aprender a predecir la próxima palabra en una secuencia de texto, dado el contexto de las palabras anteriores.

El modelo resultante después de este entrenamiento inicial se llama *modelo de lenguaje no supervisado*. En esta etapa, el modelo tiene una comprensión general del lenguaje pero no está adaptado a tareas específicas. Es decir, no podría interactuar con un usuario como lo hace ChatGPT sino que simplemente intentaría seguir prediciendo las siguientes palabras. Veamos ejemplos:

Comparación: Modelo de Lenguaje Base vs Asistente Fine-tuned

Pregunta del Usuario:

¿Cuál es la capital de Francia?

Modelo de Lenguaje Base (No Fine-tuned):

`Simplemente predice palabras...`

¿Cuál es la capital de Francia? La ciudad de... las calles... los monumentos...

Asistente Fine-tuned:

`Entiende y responde apropiadamente`

La capital de Francia es París. Es una ciudad conocida por la Torre Eiffel, el Louvre y su rica historia cultural.

Pregunta del Usuario:

¿Por qué el cielo es azul?



Modelo de Lenguaje Base (No Fine-tuned):

`Simplemente predice palabras...`

¿Por qué el cielo es azul? El cielo tiene... color... cuando... la luz...



Asistente Fine-tuned:

`Entiende y responde apropiadamente`

El cielo aparece azul debido al fenómeno de dispersión de Rayleigh: las moléculas en la atmósfera dispersan la luz solar, y dispersan más las longitudes de onda cortas (azul) que las largas.

Para llegar a ChatGPT el modelo debe pasar por el siguiente proceso:

2. Ajuste Fino (Fine-Tuning)

Después del *preentrenamiento*, el modelo no supervisado pasa por una etapa de *ajuste fino*. Este proceso adapta el modelo a tareas específicas y mejora su rendimiento en contextos relevantes.

- **Datos de *ajuste fino***: El modelo se ajusta usando un conjunto de datos más pequeño y específico, que puede incluir ejemplos de conversaciones y tareas relevantes.
- **Supervisión humana**: Los datos de *ajuste fino* suelen ser *anotados* por humanos para asegurarse de que el modelo aprenda respuestas más precisas y apropiadas.

Durante este proceso de *ajuste fino*, los parámetros del modelo se ajustan para optimizar su rendimiento en tareas específicas, como la generación de texto coherente en una conversación.



Ajuste Fino (Fine-Tuning)

Datos Específicos

Se utiliza un conjunto de datos más pequeño y específico, incluyendo ejemplos de conversaciones.

Supervisión Humana

Los datos son anotados por humanos para asegurar respuestas más precisas y apropiadas.

Optimización

Los parámetros del modelo se ajustan para mejorar su rendimiento en tareas específicas.

Proceso de Fine-tuning: De Texto Crudo a Respuestas Útiles

Tarea: Responder preguntas médicas

Datos de Pre-entrenamiento:

Libros de medicina, artículos científicos, documentación médica

Salida del modelo pre-entrenado:

Los síntomas de diabetes incluyen... la glucosa en sangre... el páncreas produce... los niveles de azúcar...

Datos de Fine-tuning:

```
P: ¿Cuáles son los síntomas de la diabetes?
```

```
R: Los síntomas principales incluyen sed excesiva, micción frecuente, fatiga y visión borrosa. Es importante consultar a un médico para un diagnóstico adecuado.
```

Respuesta del Modelo Fine-tuned:

Los síntomas principales de la diabetes incluyen: sed excesiva, micción frecuente, fatiga y visión borrosa. Si experimentas estos síntomas, te recomiendo consultar a un profesional de la salud para una evaluación adecuada. ¿Has notado alguno de estos síntomas?

3. Aprendizaje por Refuerzo (Reinforcement Learning)

Aprendizaje por Refuerzo

1 Interacción con Humanos

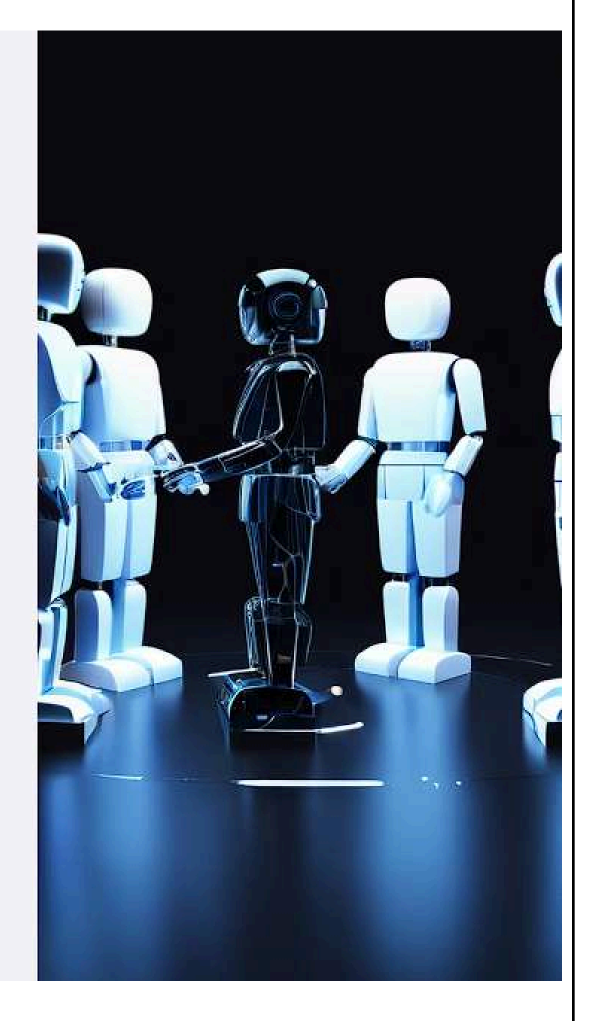
Los usuarios interactúan con el modelo y califican las respuestas según su calidad.

2 Algoritmos de Aprendizaje

Utilizan las calificaciones para ajustar los parámetros del modelo.

3 Mejora Continua

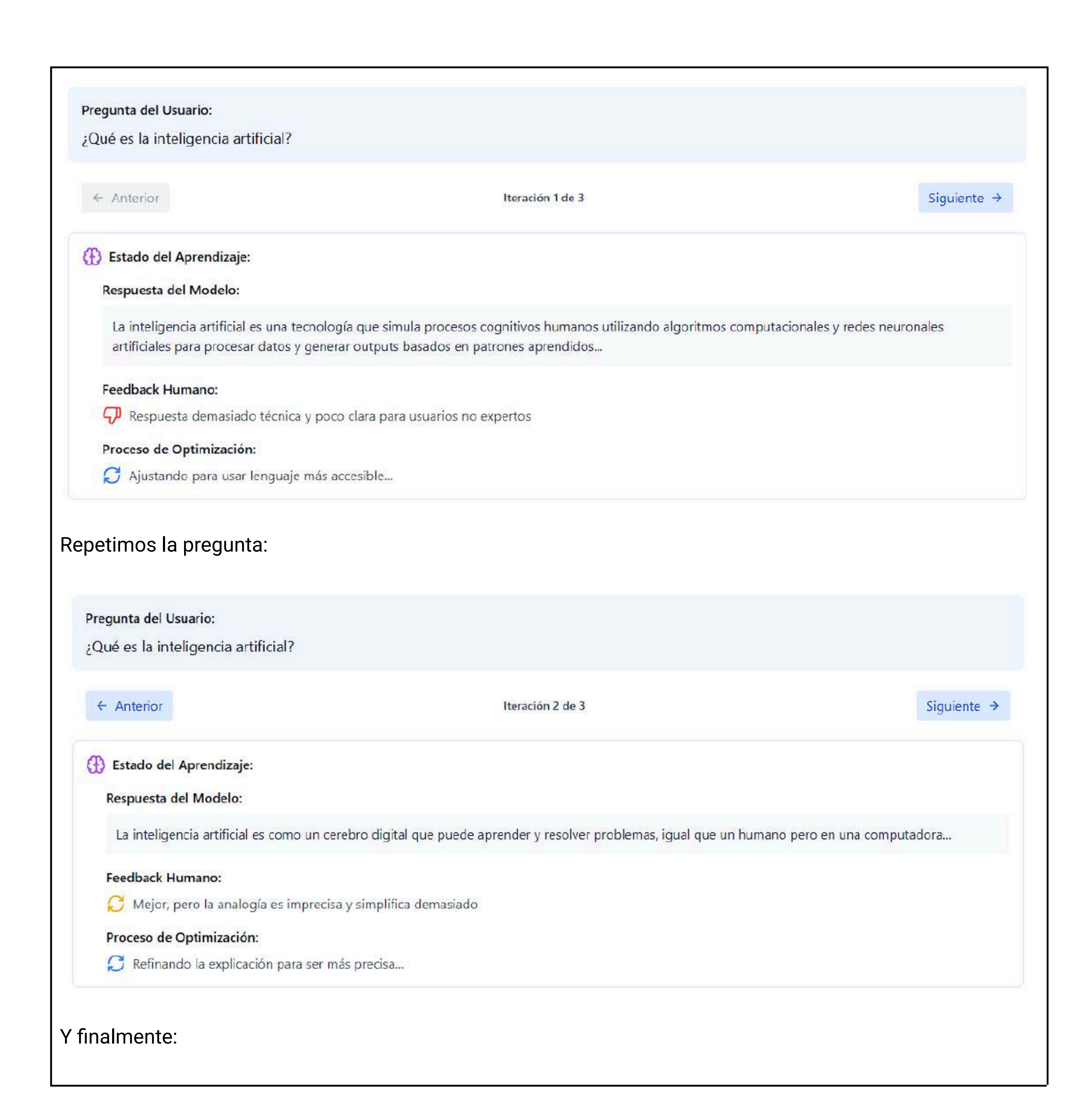
Se recompensan respuestas de alta calidad y se penalizan las de baja calidad.



El último paso en el desarrollo de ChatGPT es el uso de aprendizaje por refuerzo para mejorar la calidad de sus respuestas. Este método utiliza retroalimentación humana para ajustar aún más el modelo. Esto quiere decir que el modelo requiere de instrucción humana que le de *feedback* acerca de si su comportamiento ha sido bueno o no, ayudando al modelo a optimizar sus parámetros con el proceso de *descenso de gradiente* que hemos visto en este libro.

- **Interacción con Humanos**: Los usuarios interactúan con el modelo, y los humanos califican las respuestas del modelo en función de su calidad.
- **Algoritmos de Aprendizaje por Refuerzo**: Utilizan estas calificaciones para ajustar los parámetros del modelo, recompensando respuestas de alta calidad y penalizando las de baja calidad.

Veamos un ejemplo de cómo se utilizaría el aprendizaje por refuerzo:



Repetimos la pregunta:

Y finalmente:



En este ejemplo sobre la pregunta "¿Qué es la inteligencia artificial?", podemos ver claramente el proceso de mejora a través del aprendizaje por refuerzo. El modelo comienza dando una respuesta muy técnica llena de jerga especializada sobre "procesos cognitivos" y "algoritmos computacionales", lo que genera un feedback negativo por ser poco comprensible para usuarios no expertos. En la segunda iteración, el modelo intenta simplificar su explicación usando la analogía de un "cerebro digital", pero recibe feedback intermedio porque la analogía es imprecisa y simplifica demasiado el concepto. Finalmente, en la tercera iteración, el modelo encuentra el balance perfecto: explica la IA como "una tecnología que permite a las computadoras aprender de ejemplos y tomar decisiones, similar a cómo los humanos aprendemos de la experiencia", añadiendo ejemplos concretos que hacen la explicación clara y precisa sin perder rigor técnico.

Memorizar vs Aprender: El debate sobre cómo entrenar mejor a una IA

Como vimos anteriormente, el desarrollo de modelos como ChatGPT involucra varios pasos: un preentrenamiento inicial masivo, seguido de ajuste fino supervisado (SFT) y finalmente aprendizaje por refuerzo (RL). Recordemos el ejemplo que analizamos donde ChatGPT mejoró progresivamente su explicación sobre qué es la IA, pasando de una respuesta técnica a una más balanceada y accesible. Este proceso ilustra perfectamente el debate que ha surgido recientemente en la comunidad de IA.

Un estudio reciente de Chu et al. (2024) titulado "SFT Memorizes, RL Generalizes"[118] arroja nueva luz sobre algo que ya intuíamos al analizar el proceso de entrenamiento. Los investigadores demuestran que mientras el ajuste fino supervisado tiende a hacer que los modelos memoricen patrones específicos, el aprendizaje por refuerzo parece fomentar una comprensión más profunda.

Para entenderlo mejor, recordemos la analogía del chef que usamos al hablar de redes neuronales, pero adaptémosla a este contexto:

- El preentrenamiento sería como exponer al chef a todos los libros de cocina existentes
- El SFT sería como hacerle practicar recetas específicas una y otra vez

[118] https://huggingface.co/papers/2501.17161

- El RL sería como dejarlo experimentar en la cocina con retroalimentación sobre el resultado final

Los autores encontraron que los modelos entrenados con RL muestran una capacidad superior para adaptarse a situaciones nuevas, especialmente cuando se utilizan recompensas basadas en resultados. Sin embargo, también descubrieron algo interesante que refuerza lo que notamos en las fases de entrenamiento de ChatGPT: el SFT no es prescindible. De hecho, así como vimos que ChatGPT necesita pasar por el ajuste fino antes del RL, el estudio confirma que el SFT juega un papel crucial como paso previo, estableciendo una base sobre la cual el RL puede construir sus capacidades de generalización.

Esta investigación, respaldada por investigadores de instituciones líderes como Google Research y UC Berkeley, tiene implicaciones importantes para el futuro desarrollo de modelos de IA. Nos muestra que, para crear sistemas verdaderamente adaptables, necesitamos encontrar el equilibrio correcto entre la memorización estructurada (SFT) y el aprendizaje basado en la experiencia (RL), utilizando efectivamente ambos tipos de señales de entrenamiento para ajustar los millones de parámetros que componen estos modelos.

Conceptos clave en el funcionamiento de ChatGPT

1. **Transformers**:
    - **Arquitectura**: ChatGPT está basado en la arquitectura de transformers que ya hemos visto bastante y que utiliza mecanismos de atención para procesar y generar texto.
    - **Autoatención**: Permite al modelo considerar diferentes partes del texto de entrada simultáneamente, mejorando la coherencia y relevancia de las respuestas.
2. **Gradiente y descenso de gradiente**:
    - **Optimización**: Algoritmos como *Adam* se utilizan para ajustar los parámetros del modelo, minimizando la función de pérdida y mejorando la precisión del modelo.
3. **Redes Neuronales *Deep Learning***:
    - **Capas**: ChatGPT utiliza múltiples capas de redes neuronales para aprender representaciones complejas del lenguaje.
    - **Parámetros**: Millones de parámetros ajustables que el modelo optimiza durante el entrenamiento.

ChatGPT, ¿cómo concluirías esta sección?

***ChatGPT** es un ejemplo avanzado de cómo los modelos de lenguaje pueden transformar la generación de contenido y la interacción con la tecnología. Desde su entrenamiento inicial en grandes conjuntos de datos textuales, pasando por el ajuste fino específico, hasta el aprendizaje por refuerzo basado en retroalimentación humana, ChatGPT demuestra el poder de la inteligencia artificial en comprender y generar lenguaje natural. Al comprender los conceptos clave detrás de su funcionamiento, podemos apreciar mejor cómo esta tecnología se integra en nuestras vidas y mejora nuestras interacciones diarias.*

03. 🔍 Monitoreo y mantenimiento preventivo

Monitoreo y Mantenimiento Preventivo con IA

La inteligencia artificial revoluciona el mantenimiento industrial. Mediante sensores y análisis en tiempo real, detecta problemas antes de que ocurran.

Esto mejora la eficiencia y reduce el tiempo de inactividad. Exploremos cómo funciona este proceso innovador.



El mantenimiento preventivo es un conjunto sistemático de acciones programadas que se realizan de manera regular en equipos, maquinaria e instalaciones, antes de que ocurra una falla o avería. Este tipo de mantenimiento se basa en la inspección periódica, la limpieza, lubricación, ajustes y reemplazo de componentes que muestran desgaste, con el objetivo de prevenir problemas futuros y mantener el funcionamiento óptimo del equipo. Al seguir un calendario establecido de revisiones y servicios, se pueden identificar y corregir pequeños problemas antes de que se conviertan en fallas mayores.

La implementación de un programa de mantenimiento preventivo ofrece numerosas ventajas, incluyendo la reducción de costos operativos a largo plazo, la prolongación de la vida útil de los equipos, la disminución de tiempos de inactividad no planificados y la mejora en la seguridad del personal que opera la maquinaria. Este enfoque proactivo permite a las organizaciones mantener una producción más estable y predecible, evitando las interrupciones inesperadas que pueden resultar costosas tanto en términos de reparaciones como de pérdida de productividad. Además, el mantenimiento preventivo ayuda a mantener la eficiencia energética de los equipos y contribuye a un mejor cumplimiento de las normativas de seguridad y calidad.

¿Cómo se realiza el monitoreo y mantenimiento preventivo?

Paso 1: Recopilación de datos en tiempo real

Recopilación de Datos en Tiempo Real



Temperatura

Sensores térmicos monitorean constantemente los cambios de temperatura en equipos críticos.



Vibración

Acelerómetros detectan vibraciones anómalas que podrían indicar desgaste o desalineación.



Presión

Manómetros digitales registran fluctuaciones de presión en sistemas hidráulicos y neumáticos.



Sonido

Micrófonos ultrasensibles captan variaciones acústicas imperceptibles para el oído humano.

Primero, una vez más, comenzamos por la recopilación de datos. En este caso es importante acercarse lo más posible al tiempo real por ejemplo con sensores instalados en equipos industriales o sistemas de transporte. Estos datos pueden incluir:

- Temperaturas
- Vibración
- Presión
- Niveles de aceite
- Sonido

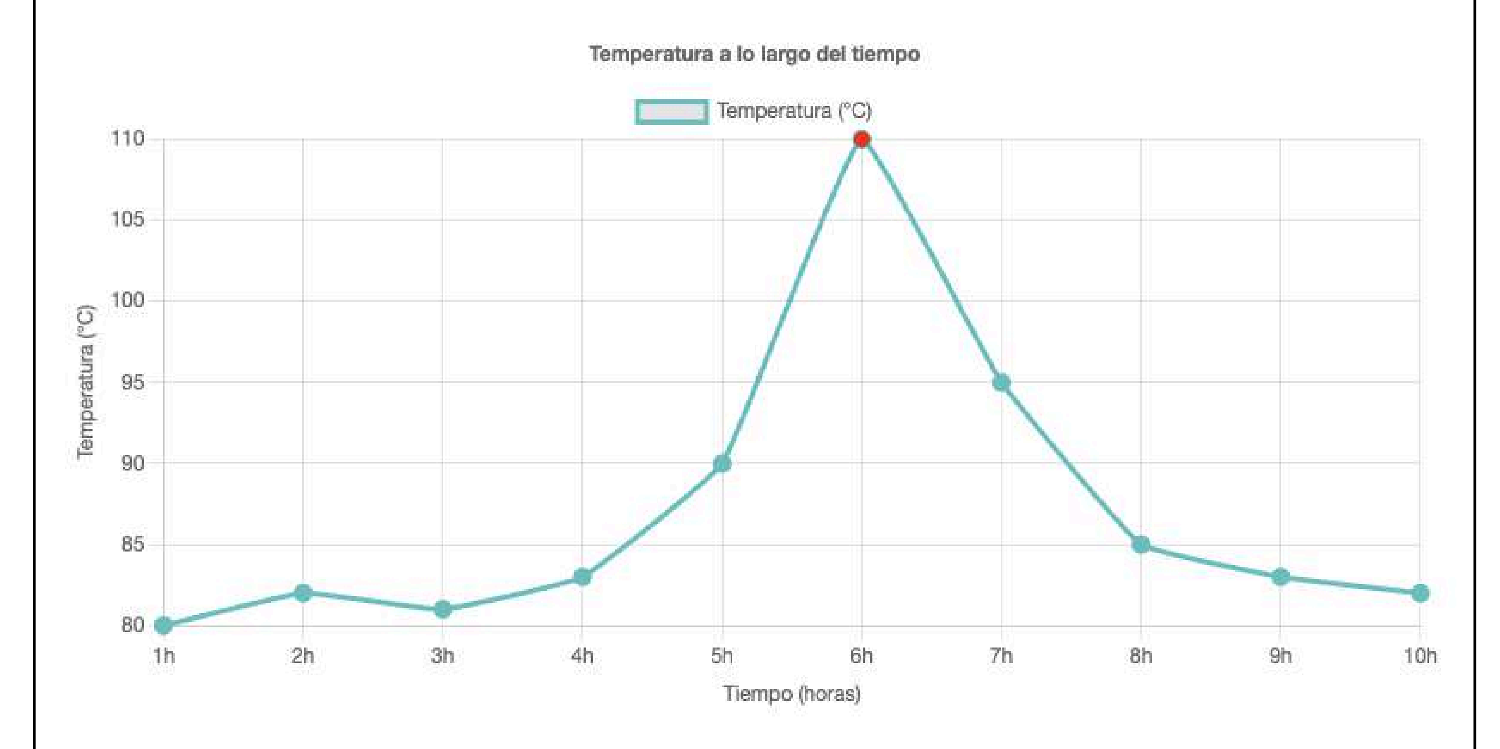
Acá vemos un posible ejemplo de ellos:

Datos en Tiempo Real

Temperatura:	85°C
Vibración:	2.5 mm/s
Presión:	5.2 bar
Nivel de aceite:	80%
Nivel de sonido:	72 dB

Paso 2: Análisis de datos con modelos predictivos

Se utilizan modelos de IA para analizar estos datos, establecer el comportamiento *normal* y entonces así poder detectar patrones que indiquen un posible funcionamiento *anómalo*. El escenario ideal es donde este comportamiento anómalo se detecta con suficiente anticipación para que no se sufran sus consecuencias y se pueda prevenir o tomar la acción menos costosa. Como los datos nos llegan en tiempo real y en secuencia temporal, podemos considerarlos como *series temporales*.



Técnicas como el análisis de series temporales y el *machine learning* son esenciales para este propósito.

1. Análisis de series temporales

Modelos ARIMA (AutoRegressive Integrated Moving Average)

Los modelos ARIMA son una herramienta estadística diseñada específicamente para el análisis y predicción de series temporales. Su efectividad radica en tres componentes principales:

- **Componente Autorregresivo (AR)**: Analiza cómo los valores pasados influyen en los valores futuros. Por ejemplo, en una máquina industrial, las lecturas de temperatura de las últimas 24 horas pueden ayudar a predecir la temperatura en la próxima hora.

- **Componente Integrado (I)**: Se encarga de hacer la serie temporal *estacionaria*, eliminando tendencias y ciclos[119]. Esto es crucial cuando trabajamos con datos que naturalmente tienden a aumentar o disminuir con el tiempo, como el desgaste gradual de una pieza.
- **Componente de Media Móvil (MA)**: Aprende de los errores de predicción recientes para ajustar predicciones futuras. Por ejemplo, si un sensor de presión constantemente predice por debajo del valor real, el componente MA detectará este patrón de error y ajustará automáticamente las siguientes predicciones al alza.

Aplicaciones prácticas:

- 🔄 **Predicción de fallos en rodamientos basados en patrones de vibración**: Los sensores de vibración en un rodamiento industrial generan datos continuos que, analizados con un modelo como ARIMA, pueden revelar patrones sutiles antes de un fallo. Por ejemplo, en una turbina eólica, el modelo podría detectar un incremento gradual en las vibraciones a ciertas frecuencias que indican el inicio de un desgaste en la pista externa del rodamiento, permitiendo programar el mantenimiento antes de que ocurra una falla catastrófica.
- ⏳ **Estimación de vida útil restante de componentes**: En una línea de producción de automóviles, los robots de soldadura tienen electrodos que se desgastan con el uso. Un modelo ARIMA puede analizar la tendencia de la calidad de los puntos de soldadura a lo largo del tiempo y predecir cuándo el electrodo alcanzará un umbral crítico de desgaste. Por ejemplo, si la resistencia eléctrica del electrodo muestra un patrón de degradación consistente, el modelo puede estimar que quedan 2,500 puntos de soldadura antes de requerir reemplazo.
- 🔧 **Planificación de intervalos de mantenimiento optimizados**: En una flota de camiones de transporte, un modelo ARIMA puede analizar el histórico de fallos, consumo de combustible y rendimiento del motor para optimizar los intervalos de mantenimiento. Por ejemplo, en lugar de realizar mantenimientos cada 10,000 km de manera fija, el modelo podría determinar que algunos camiones que operan en rutas montañosas necesitan mantenimiento cada 8,000 km, mientras que los que operan en autopistas planas pueden extenderlo hasta 12,000 km, optimizando costos sin comprometer la seguridad.

Redes Neuronales Recurrentes (RNN)

Las RNN son arquitecturas de *deep learning* especialmente diseñadas para procesar secuencias de datos. Su característica distintiva es la capacidad de mantener una "memoria" de eventos pasados:

- **Memoria a corto y largo plazo (LSTM)**: Una variante especial de RNN que puede recordar patrones tanto recientes como antiguos. Esto es crucial para detectar degradación gradual en equipos.
- **Arquitectura Bidireccional**: Analiza las secuencias de datos tanto hacia adelante como hacia atrás en el tiempo, permitiendo capturar relaciones más complejas.

---

[119] En el análisis de series temporales, la tendencia representa el comportamiento general o dirección a largo plazo que sigue una variable en el tiempo, pudiendo ser creciente, decreciente o estable, mientras que un ciclo se refiere a las oscilaciones o fluctuaciones que se repiten de manera periódica alrededor de la tendencia, con una duración superior a un año. Mientras la tendencia refleja cambios estructurales o evolutivos (como el crecimiento poblacional de un país), los ciclos no necesariamente son regulares en su duración o amplitud y suelen estar asociados a fases alternantes (como los ciclos económicos de auge y recesión), distinguiéndose de las variaciones estacionales que ocurren dentro del mismo año.

Casos de uso:

- 🚨 **Detección de anomalías en tiempo real**: En una planta química, una RNN puede monitorear simultáneamente docenas de sensores (temperatura, presión, flujo, pH) y detectar patrones anómalos que serían invisibles al analizar cada sensor por separado. Por ejemplo, una combinación específica de pequeñas variaciones en múltiples parámetros podría indicar el inicio de una reacción química no deseada, permitiendo una intervención temprana.
- 📊 **Predicción de fallos basada en múltiples sensores**: En un aerogenerador moderno, una RNN puede integrar datos de sensores de vibración, temperatura, velocidad del viento, ángulo de las palas y producción eléctrica. El modelo aprende las interrelaciones complejas entre estas variables y puede predecir fallos que surgen de la interacción entre componentes. Por ejemplo, podría detectar que una combinación específica de vibraciones y temperaturas, sólo bajo ciertas condiciones de viento, indica un problema emergente en la caja de engranajes.
- 📉 **Análisis de patrones de degradación complejos**: En una línea de producción de semiconductores, una RNN puede analizar la degradación de la calidad del producto a lo largo de múltiples etapas de fabricación. El modelo puede identificar patrones sutiles de degradación que se desarrollan a través de diferentes escalas temporales, como el desgaste gradual de las herramientas de grabado combinado con variaciones en la calidad de los materiales entrantes y fluctuaciones en las condiciones ambientales.

2. Aprendizaje Supervisado

Regresión Lineal y sus Variantes

La regresión lineal como ya vimos antes, aunque simple, es sorprendentemente efectiva para muchas aplicaciones de mantenimiento predictivo:

- **Regresión Lineal Simple**: Modela la relación entre una variable independiente (como el tiempo de uso) y una variable dependiente (como el desgaste). Especialmente útil cuando tenemos una hipótesis inicial de cuál es la variable más importante para predecir el suceso.
- **Regresión Lineal Múltiple**: Incorpora múltiples variables predictoras, como temperatura, presión, vibración, etc. Ideal cuando queremos estudiar el efecto de todas las variables posibles y además no tenemos una hipótesis inicial que plantear.

Aplicaciones:

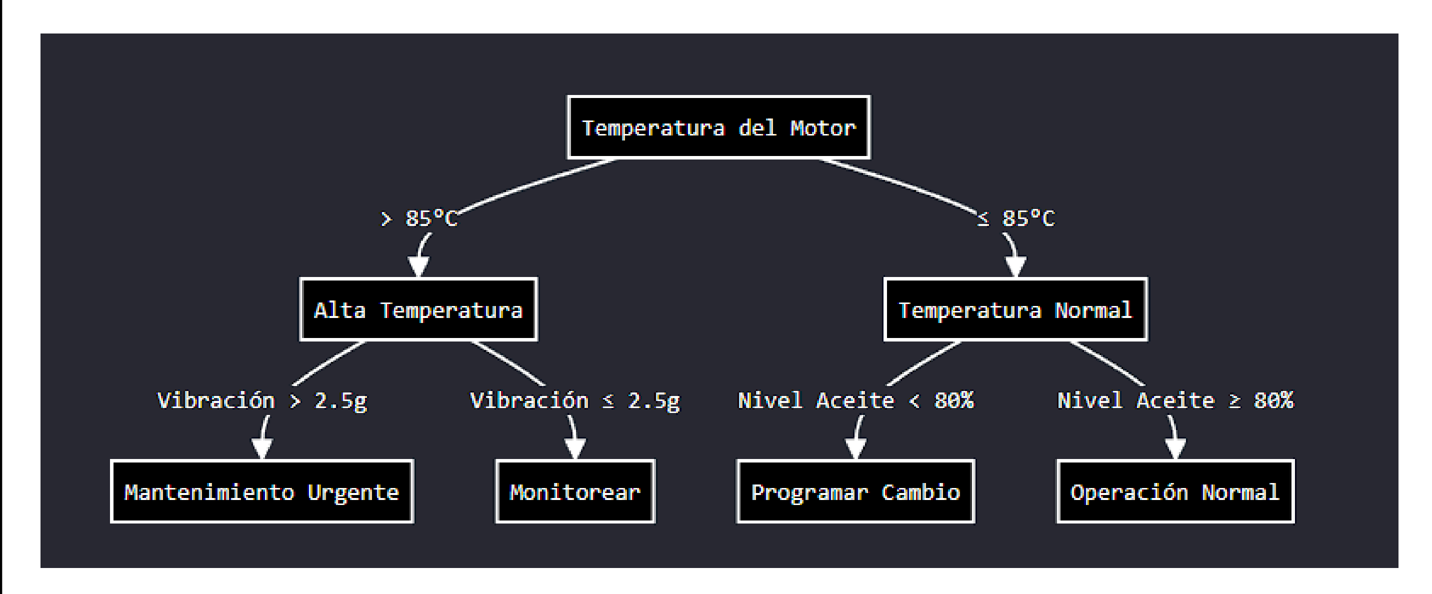
- ⏳ **Predicción de vida útil restante**: En una batería industrial de un sistema de alimentación ininterrumpida (UPS), la regresión lineal puede analizar la tendencia de la capacidad de carga a lo largo del tiempo. Utilizando variables como el número de ciclos de carga/descarga, temperatura promedio de operación y profundidad de descarga típica, el modelo puede proyectar cuándo la capacidad caerá por debajo del umbral mínimo aceptable, permitiendo programar reemplazos proactivos.
- 💰 **Estimación de costos de mantenimiento**: En una flota de maquinaria pesada, la regresión lineal múltiple puede predecir los costos futuros de mantenimiento basándose en factores como edad del equipo, horas de operación, condiciones ambientales y historial de reparaciones. Por ejemplo, el modelo podría determinar que cada año de antigüedad adicional incrementa los costos de mantenimiento en un 15%, mientras que operar en ambientes polvorientos añade un 20% adicional.

- ⚡ **Optimización de intervalos de servicio**: En transformadores eléctricos de alta tensión, la regresión lineal puede analizar la degradación del aceite aislante considerando factores como la carga promedio, temperatura ambiente y niveles de humedad. El modelo puede determinar el momento óptimo para el mantenimiento preventivo, equilibrando el costo del servicio con el riesgo de fallo, por ejemplo, sugiriendo intervalos más cortos para transformadores en zonas húmedas y cálidas.

Árboles de Decisión y Random Forest

También los hemos visto ya en este libro. Estos modelos son excelentes para clasificar estados de equipos y tomar decisiones de mantenimiento:

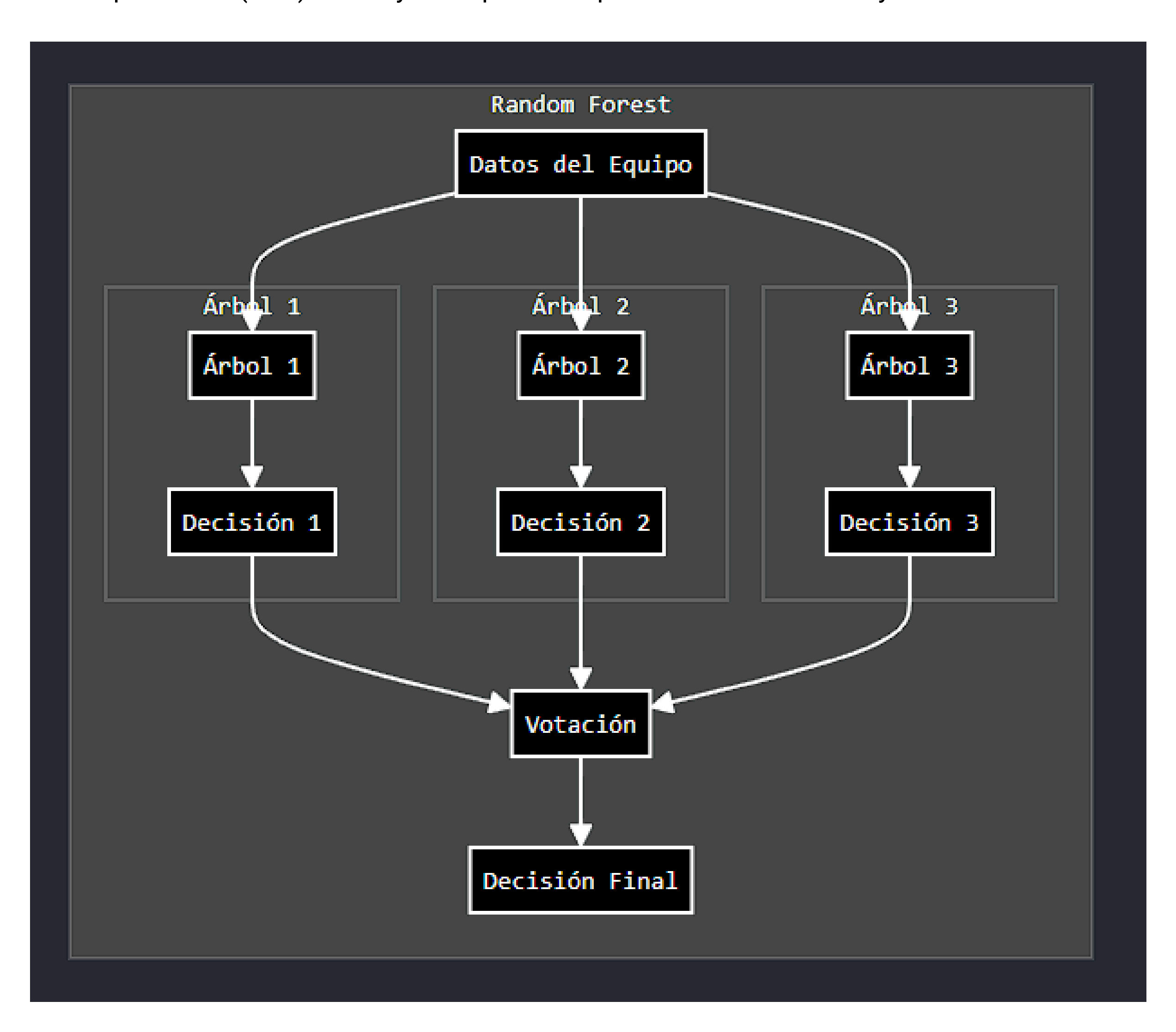
- **Árboles de Decisión**:
  - **Fáciles de interpretar y visualizar**: En el caso de un motor industrial, el árbol crea un flujo de decisión intuitivo que comienza con una simple pregunta sobre la temperatura (¿> 85°C?). Cualquier técnico puede seguir estas ramas visuales para llegar a una decisión de mantenimiento, desde "Operación Normal" hasta "Mantenimiento Urgente", sin necesidad de entender algoritmos complejos.
  - **Pueden manejar tanto datos numéricos como categóricos**: El árbol combina mediciones numéricas precisas (temperatura: 85°C, vibración: 2.5g) con variables categóricas derivadas (Alta Temperatura/Temperatura Normal, Nivel de Aceite Alto/Bajo). Esta flexibilidad permite crear un modelo completo que aprovecha todos los tipos de datos disponibles del motor.
  - **Excelentes para crear reglas de decisión claras**: El árbol genera reglas de acción específicas como "Si Temperatura > 85°C Y Vibración > 2.5g, entonces realizar Mantenimiento Urgente" o "Si Temperatura ≤ 85°C Y Nivel de Aceite ≥ 80%, entonces continuar con Operación Normal". Estas reglas son directamente implementables en el sistema de monitoreo del motor.



- **Random Forest (Bosques Aleatorios)**:
  - **Combinan múltiples árboles para mayor precisión**: En el monitoreo de un grupo de motores similares, cada árbol podría analizar diferentes subconjuntos de datos. Por ejemplo, mientras un árbol se enfoca en la relación temperatura-vibración, otro podría centrarse en el nivel de

aceite-ruido, y un tercero en carga-consumo eléctrico. El Random Forest combina todas estas "opiniones" para tomar una decisión más robusta sobre el estado del motor.

  - **Reducen el sobreajuste**: Si un solo árbol de decisión se enfocara demasiado en un patrón específico (como una lectura anómala de temperatura que ocurrió una vez), el Random Forest lo compensa con los otros árboles. Por ejemplo, si un motor tuvo un pico de temperatura debido a un día excepcionalmente caluroso, el modelo no sobrereaccionará a temperaturas elevadas en el futuro si los otros indicadores (vibración, nivel de aceite) están normales.
  - **Proporcionan medidas de importancia de variables**: El Random Forest puede determinar qué variables son más cruciales para predecir fallos en el motor. Podría revelar, por ejemplo, que la vibración es el indicador más confiable de problemas inminentes (importancia del 40%), seguido por la temperatura (30%), mientras que el nivel de aceite tiene menor impacto predictivo (15%). Esto ayuda a priorizar qué sensores mantener y monitorear más de cerca.



Usos prácticos:

- 🔍 **Clasificación de estados de equipos**: En una planta de manufactura con múltiples motores eléctricos, un Random Forest puede clasificar continuamente el estado de cada motor en categorías como "Óptimo", "Requiere Atención" o "Crítico". El modelo toma en cuenta todas las variables (temperatura, vibración, nivel de aceite) y asigna un estado basado en el "voto mayoritario" de todos

sus árboles. Por ejemplo, si 80 de 100 árboles clasifican un motor como "Requiere Atención" basándose en diferentes combinaciones de variables, ese será el diagnóstico final.

- 🛠️ **Diagnóstico de fallos**: Cuando un motor presenta problemas, el Random Forest puede analizar sus patrones de fallo comparándolos con casos históricos. Si históricamente una combinación de alta temperatura (>85°C) y vibración elevada (>2.5g) ha llevado a fallos en el rodamiento en el 90% de los casos, el modelo puede diagnosticar este problema específico con alta confianza, permitiendo una reparación dirigida.
- ⚡ **Priorización de mantenimiento**: En una instalación con docenas de motores, el Random Forest puede crear un ranking de urgencia de mantenimiento basado en múltiples factores. Por ejemplo, un motor con temperatura elevada pero baja vibración podría recibir una prioridad media (monitoreo), mientras que otro con alta temperatura Y alta vibración recibiría prioridad máxima (mantenimiento urgente), optimizando así la asignación de recursos de mantenimiento.

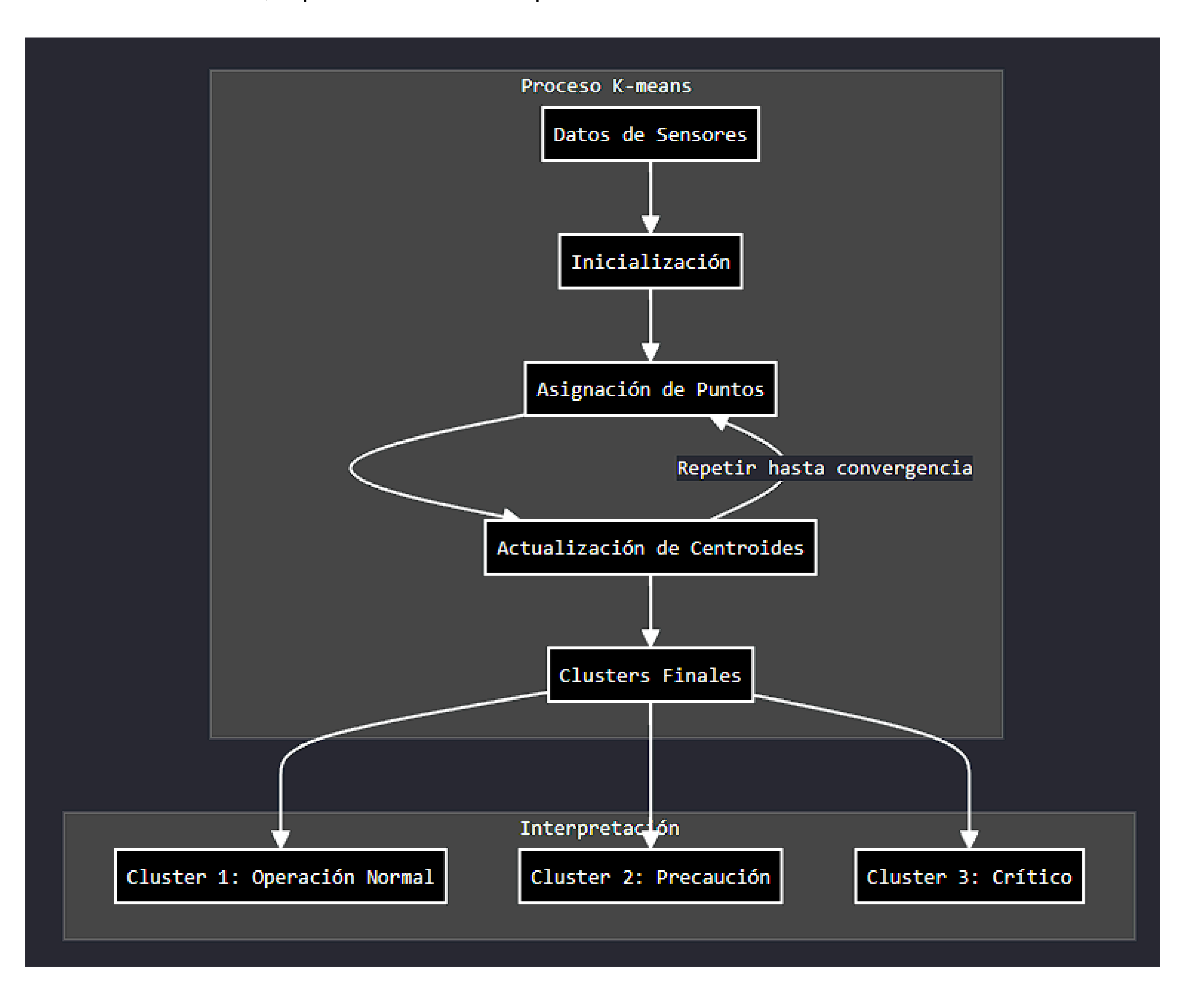
3. Aprendizaje No Supervisado

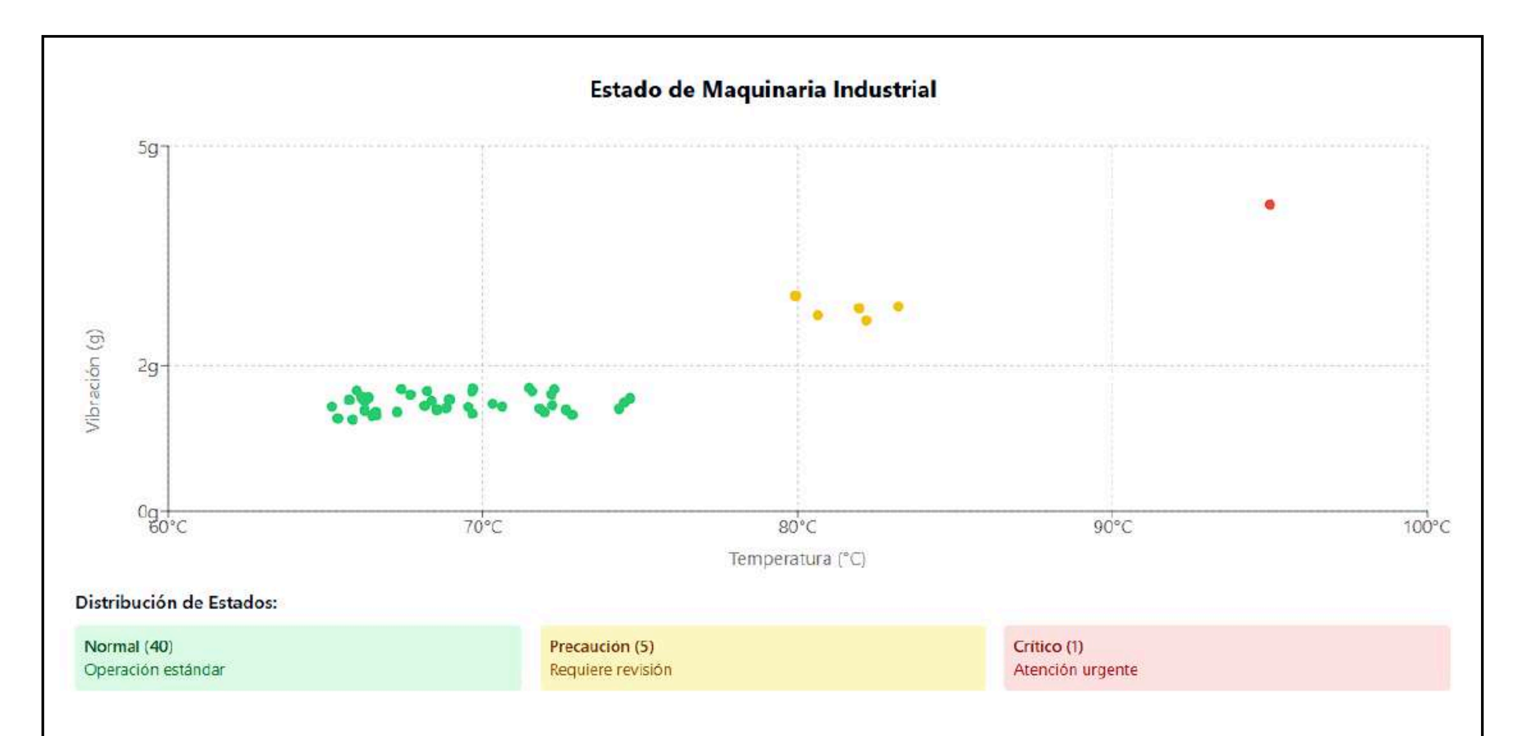
Algoritmos de Clustering (K-means)

El clustering que también hemos estudiado es fundamental para detectar patrones y anomalías sin necesidad de datos etiquetados:

- **K-means Básico**:
    - **Agrupa datos similares en clusters**: En un sistema de compresores industriales, K-means puede analizar datos históricos de presión, temperatura y consumo eléctrico para identificar patrones naturales de operación. Por ejemplo, podría detectar tres clusters principales: "operación a baja carga" (presión baja, temperatura baja, bajo consumo), "operación normal" (valores medios en todos los parámetros), y "operación en sobrecarga" (presión alta, temperatura alta, alto consumo). Esto establece una línea base para detectar desviaciones.
    - **Útil para identificar estados de operación normales vs. anómalos**: Monitoreando una turbina de gas, K-means puede establecer automáticamente qué combinaciones de variables (velocidad de rotación, temperatura de gases de escape, vibración) representan un funcionamiento normal. Si los nuevos datos comienzan a alejarse significativamente de estos clusters normales, por ejemplo, mostrando alta temperatura con baja velocidad, el sistema puede alertar sobre una posible anomalía.
    - **Puede procesar grandes volúmenes de datos eficientemente**: En una planta con cientos de sensores recolectando datos cada segundo, K-means puede procesar rápidamente millones de puntos de datos para crear un "mapa" de estados operativos. Por ejemplo, puede analizar un año completo de datos de múltiples máquinas (más de 31 millones de lecturas) y clasificarlos en patrones significativos en cuestión de minutos, algo que sería imposible de hacer manualmente.
- **Variantes Avanzadas**:
    - **DBSCAN (Density-Based Spatial Clustering of Applications with Noise)**: En el monitoreo de una flota de vehículos industriales, DBSCAN puede identificar patrones de deterioro que no siguen formas circulares o esféricas. Por ejemplo, podría detectar que el desgaste de los frenos sigue un patrón irregular que combina el peso de la carga, la pendiente de la ruta y la temperatura de operación, identificando incluso casos atípicos que otros algoritmos podrían pasar por alto.

- **Clustering Jerárquico**: En una planta petroquímica, este método puede crear una estructura tipo árbol de patrones operativos. En el nivel superior, podría distinguir entre "operación normal" y "operación anormal", luego subdividir la operación anormal en categorías más específicas como "problemas de catalizador", "problemas de temperatura" y "problemas de presión", y así sucesivamente hasta llegar a causas raíz específicas. Esto crea una taxonomía natural de estados operativos.
- **Clustering Espectral**: En turbinas eólicas, donde las relaciones entre variables son altamente no lineales, este método puede descubrir patrones complejos entre velocidad del viento, ángulo de las palas, producción de energía y vibraciones. Por ejemplo, podría identificar condiciones sutiles que llevan a una menor eficiencia antes de que se conviertan en fallos evidentes, capturando relaciones que los métodos tradicionales no detectarían.





Mira bien el gráfico de arriba. Nos muestra cómo el clustering puede clasificar automáticamente el estado de una máquina industrial basándose en dos variables clave: temperatura y vibración. Los datos se agrupan naturalmente en tres clusters distintos: un grupo grande de 40 puntos 🟢 que representa la operación normal (65-75°C, 15-20g de vibración), un grupo más pequeño de 5 puntos 🟡 que indica estado de precaución (80-85°C, 25-30g de vibración) que requiere revisión, y un punto aislado 🔴 que señala una condición crítica (95°C, 45g de vibración) que necesita atención urgente. Esta visualización permite a los operadores identificar rápidamente desviaciones del comportamiento normal y tomar acciones preventivas antes de que la máquina alcance un estado crítico.

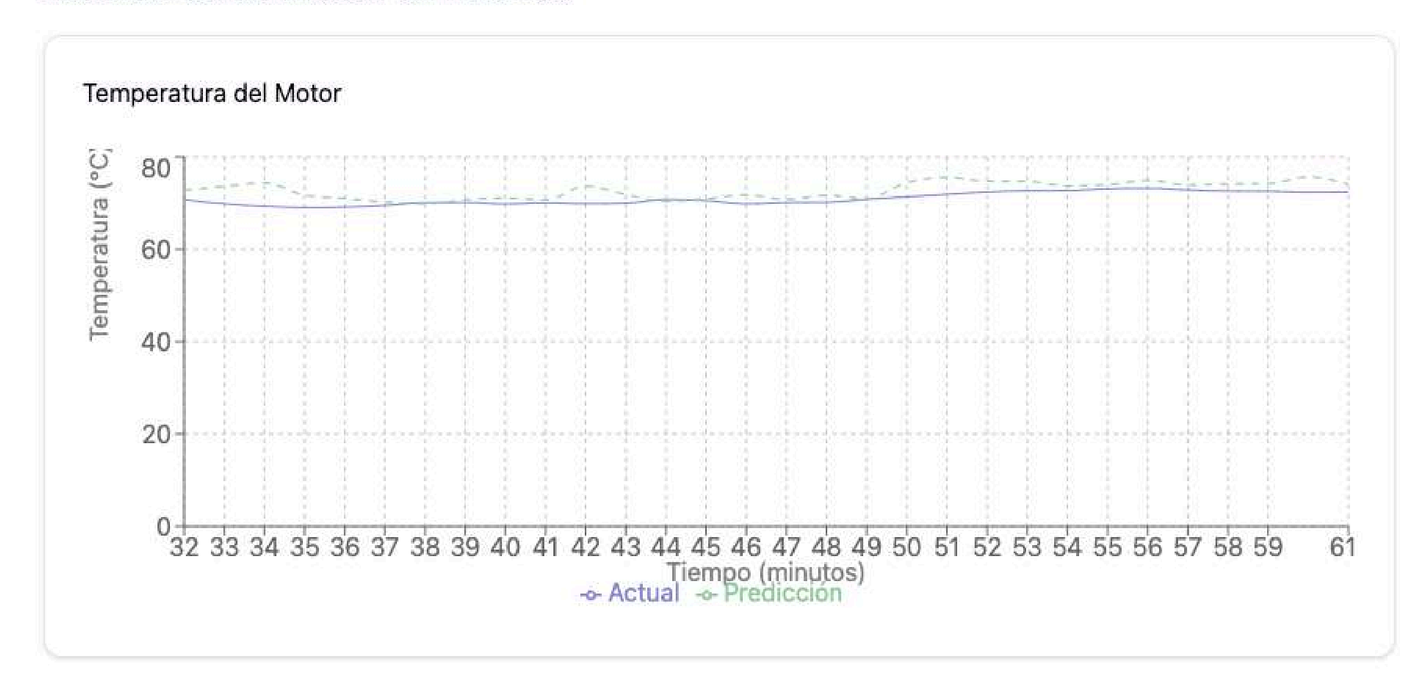
Ejemplo

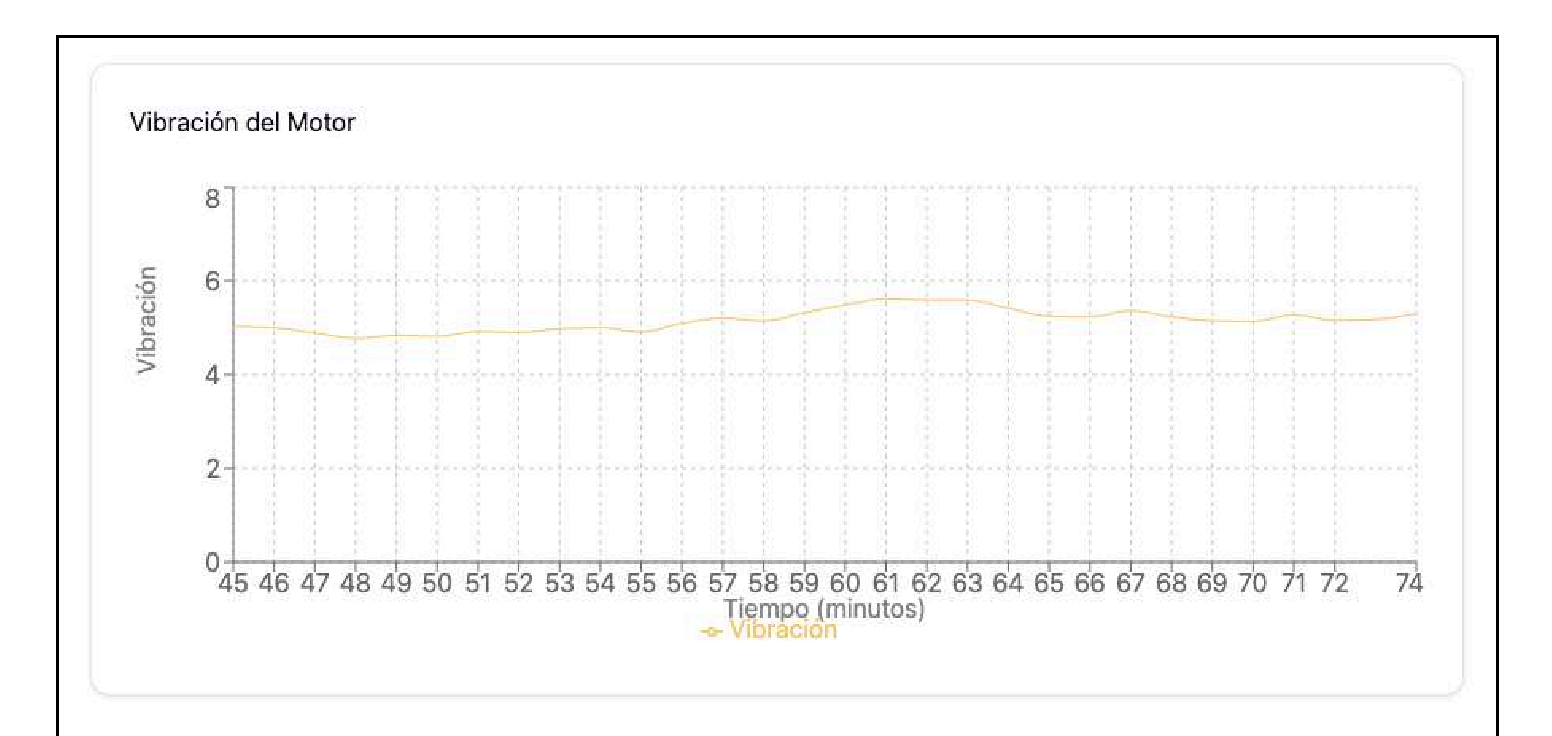
Imaginemos que estamos monitoreando un motor en una fábrica. Recopilamos datos de los sensores que miden la *temperatura* y la *vibración del motor* cada minuto.

- **Temperatura**: Se espera que la temperatura del motor se mantenga dentro de un rango específico. Un aumento inusual podría indicar un problema de refrigeración.
- **Vibración**: Niveles altos de vibración podrían indicar un desequilibrio o desgaste en los componentes del motor.

Veamos un par de gráficos que muestren un comportamiento típico de cada uno de ellos:

Monitoreo de Motor en Fábrica



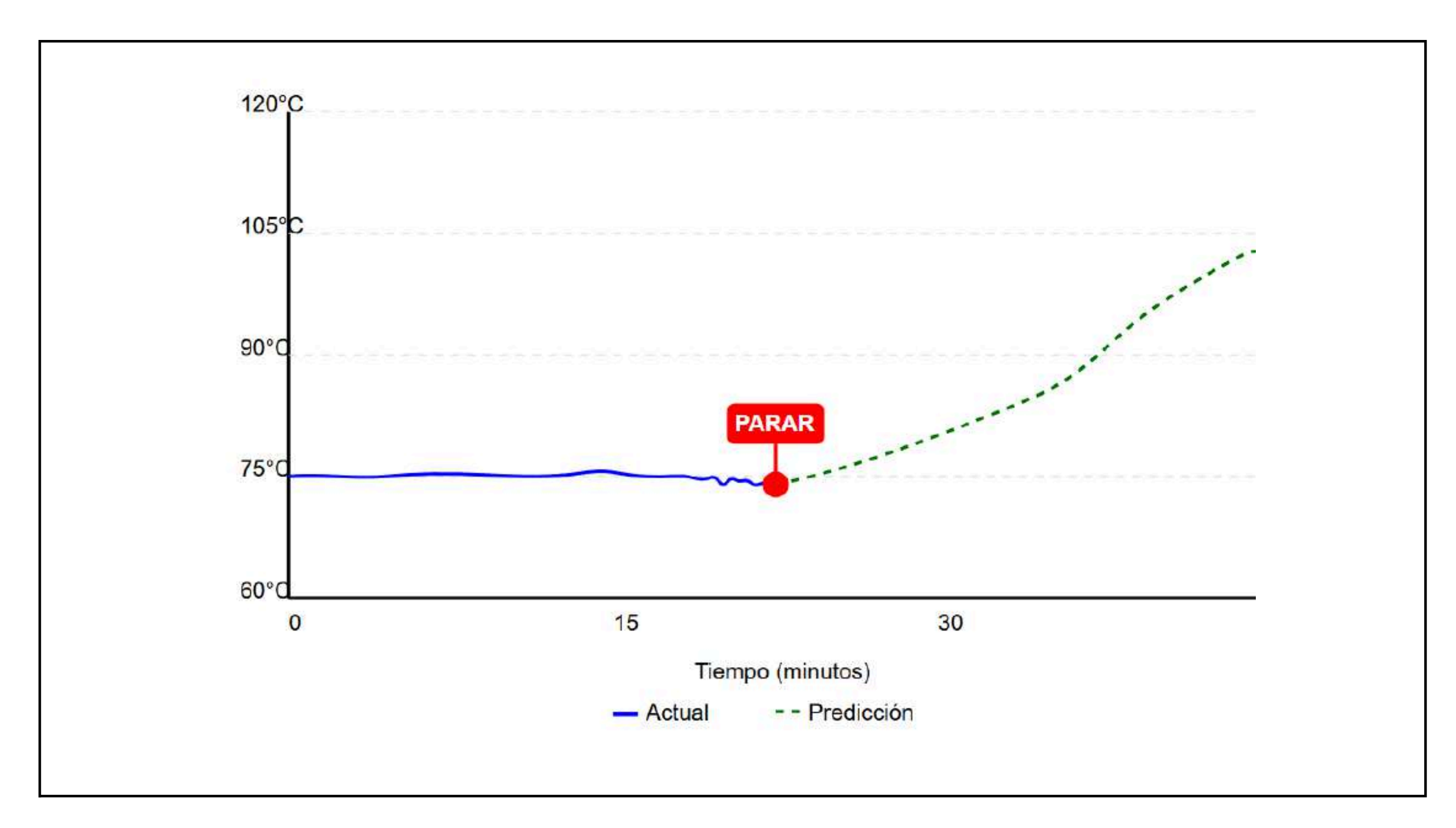


Para detectar anomalías en la temperatura del motor, hemos establecido un modelo que analiza tanto los patrones históricos como las variaciones normales de temperatura durante su funcionamiento. El modelo fue entrenado con datos de operación normal y aprendió que, en condiciones estándar, la temperatura debe mantenerse entre 70°C y 85°C. Cuando el modelo predice que la temperatura superará los 95°C en los próximos minutos, se considera una anomalía y se genera una alerta, ya que estas temperaturas están fuera del rango de operación segura del motor. Esta predicción temprana nos permite tomar medidas correctivas antes de que el motor alcance temperaturas potencialmente dañinas.

Si el modelo predice una temperatura anómala en el futuro cercano, se genera una alerta para el mantenimiento preventivo.

En el siguiente ejemplo el modelo predice un aumento anormal de temperatura debido a que ha detectado un patrón característico en las pequeñas fluctuaciones recientes que se asemeja a casos históricos de fallos en el sistema de refrigeración por aceite. Aunque la temperatura actual está dentro del rango normal, el modelo ha aprendido que cuando se observa este patrón específico de variaciones (como los ligeros aumentos y descensos que vemos en los últimos puntos de la línea azul), frecuentemente indica una fuga gradual de aceite que, una vez alcanza un nivel crítico, provocará un rápido aumento de temperatura en los próximos minutos (como muestra la línea verde punteada). Esta detección temprana permite al equipo de mantenimiento intervenir antes de que se produzca un sobrecalentamiento real del motor.

 ¡Alerta! Se predice una temperatura anormal en el futuro cercano.



El mantenimiento preventivo basado en datos y *machine learning* representa un avance significativo respecto a los métodos tradicionales de mantenimiento programado. Como hemos visto en este ejemplo, un modelo entrenado con datos históricos puede detectar patrones sutiles en el comportamiento de los equipos (como las pequeñas variaciones en la temperatura del motor) que preceden a fallos graves. Esta capacidad de predicción temprana permite intervenir en el momento óptimo: ni demasiado pronto, lo que supondría un desperdicio de recursos, ni demasiado tarde, cuando el daño ya está hecho. En nuestro caso, el sistema identificó un patrón característico de fallo inminente en el sistema de refrigeración y generó una alerta para detener el motor antes de que alcance temperaturas críticas, evitando así costes mayores de reparación y tiempo de inactividad no planificado.

04. 🚫 Detección de fraude

En nuestra sección sobre probabilidad y estadística ya vimos como tan solo con entender esos conceptos podemos aplicarlos en algo tan complejo como la detección de actividad fraudulenta. En esta sección vamos a volver a repetir un ejemplo similar pero lo vamos a hacer con mucho más nivel de detalle y modelos de IA más avanzados.



Detección de Fraude con Inteligencia Artificial

La detección de fraude financiero ha evolucionado significativamente con la implementación de la Inteligencia Artificial. Esta tecnología permite analizar patrones de transacciones en tiempo real, protegiendo a los usuarios de actividades fraudulentas.

Exploraremos cómo funciona este proceso y las técnicas de IA utilizadas.

<u>¿Cómo se realiza la detección de fraude? (¡una vez más!)</u>

Paso 1: Recopilación de datos

Primero, y para sorpresa de nadie, se recopilan datos de transacciones financieras. Estos datos pueden incluir:

- Monto de la transacción
- Ubicación de la transacción
- Hora y fecha de la transacción
- Método de pago
- Historial de transacciones del usuario

Veamos una simulación de estos datos:

Ejemplos de Datos Recopilados en Transacciones Financieras

ID Transacción	Monto	Moneda	Fecha y Hora	Método de Pago	Tipo de Tarjeta	País de Emisión	País de Transacción	IP del Cliente	Dispositivo	Categoría de Compra	Comerciante
TRX123456	$500.00	USD	2024-09-14 15:30:22	Tarjeta de Crédito	Visa	España	EE.UU.	203.0.113.45	iPhone	Electrónica	TechStore Inc.
TRX123457	€1,200.00	EUR	2024-09-14 08:45:10	PayPal	N/A	Alemania	Francia	198.51.100.67	Android	Viajes	EuroTravel Agency
TRX123458	£50.00	GBP	2024-09-13 22:15:37	Tarjeta de Débito	Mastercard	Reino Unido	Reino Unido	192.0.2.88	PC	Restaurante	London Pub Ltd.



Recopilación y Análisis de Datos

1. **Recopilación de Datos**

   Se obtienen datos de transacciones financieras, incluyendo montos, ubicaciones, fechas y métodos de pago.

2. **Preprocesamiento**

   Los datos se limpian y normalizan para su análisis posterior.

3. **Análisis de Patrones**

   Modelos de IA analizan los datos para detectar patrones indicativos de fraude.

Paso 2: Creación de características (features) para detectar patrones sospechosos

En ciencia de datos llamamos *característica* a un dato que consideramos de *buena utilidad*[120] para el objetivo de nuestro modelo. Los datos que se extraen de las diversas fuentes en casi el 100% de los casos no están ni preparados ni optimizados para una buena aplicación del *machine learning*. El trabajo del científico de datos o el ingeniero de IA es convertir los datos en *características*. En el caso de la detección de fraude, la creación de características es crucial para mejorar la detección de patrones sospechosos.

En el siguiente ejemplo podés imaginar que partimos de una base de datos de todas las transacciones individuales realizadas por clientes de un banco. El objetivo del científico de datos será resumir las transacciones en una fila por cliente, con el objetivo de establecer cuál es la actividad típica/normal de cada uno de ellos. Tenés que pensar esto como que el sistema aprenderá cómo transaccionás vos para detectar el eventual caso de que no seas vos quién ha hecho una transacción a tu nombre.

Detección de Fraude: Conversión de Transacciones a Características

Tabla de Conversión: Transacciones a Características

Esta tabla muestra cómo las transacciones individuales se convierten en características agregadas para el análisis de fraude.

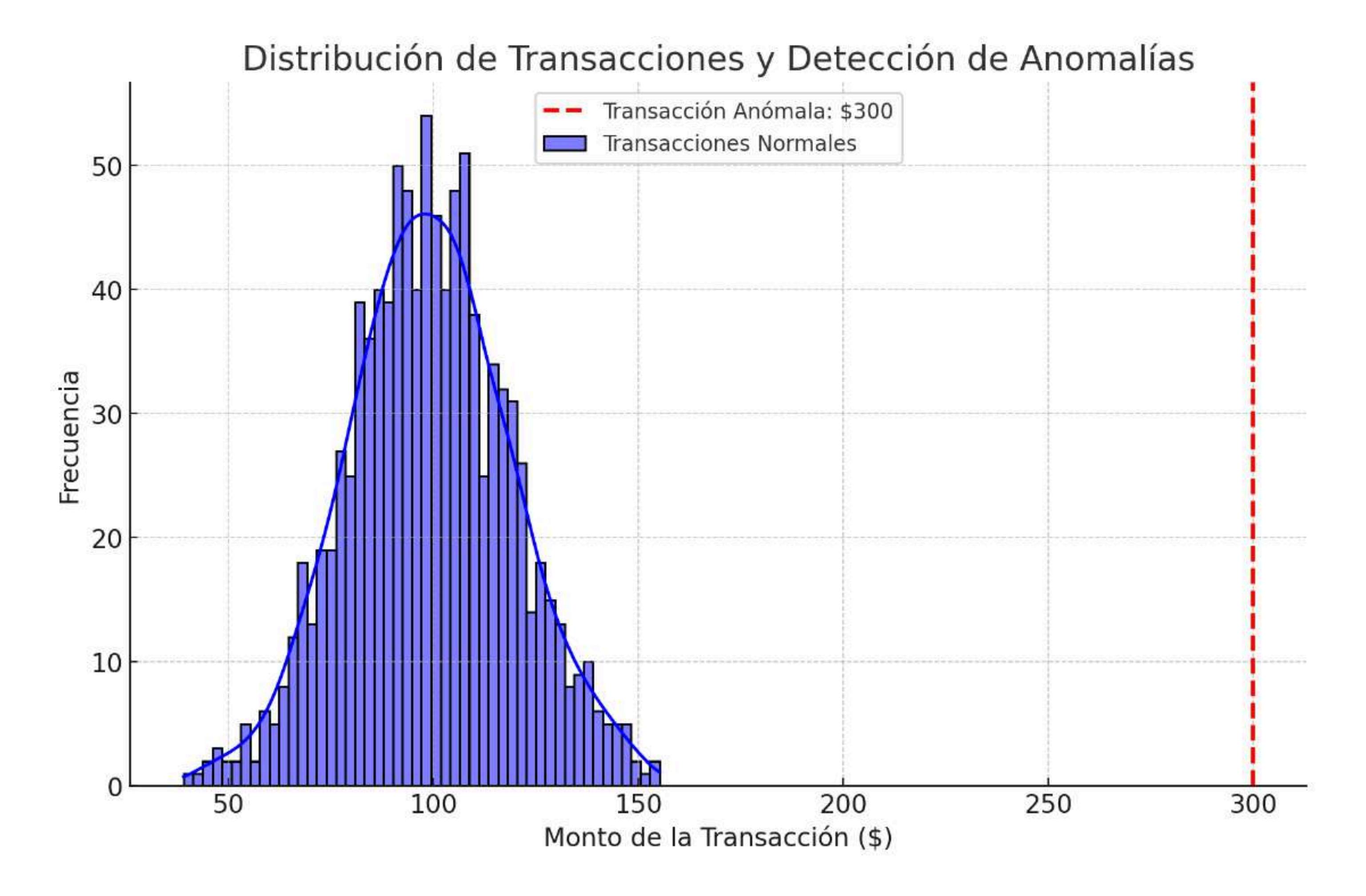
ID Usuario	Número de Transacciones (24h)	Monto Total (24h)	Promedio de Transacción	Máximo Monto de Transacción	Número de Países Distintos	Número de Dispositivos Distintos	Transacciones Fuera de Horario Habitual	Variación en Montos (%)	Velocidad de Transacciones (por hora)
User001	8	$1,245.00	$155.63	$500.00	2	3	2	78%	0.33
User002	3	$3,500.00	$1,166.67	$2,000.00	1	1	0	25%	0.13
User003	15	$750.00	$50.00	$100.00	3	2	5	40%	0.63

Explicación de las Características

- **Número de Transacciones (24h):** Total de transacciones realizadas en las últimas 24 horas.
- **Monto Total (24h):** Suma total de los montos de todas las transacciones en las últimas 24 horas.
- **Promedio de Transacción:** Monto promedio por transacción en las últimas 24 horas.
- **Máximo Monto de Transacción:** La transacción de mayor valor en las últimas 24 horas.
- **Número de Países Distintos:** Cantidad de países diferentes desde donde se realizaron transacciones.
- **Número de Dispositivos Distintos:** Cantidad de dispositivos diferentes utilizados para las transacciones.
- **Transacciones Fuera de Horario Habitual:** Número de transacciones realizadas fuera del horario típico del usuario.
- **Variación en Montos (%):** Porcentaje de variación entre el monto más alto y más bajo de las transacciones.
- **Velocidad de Transacciones (por hora):** Promedio de transacciones realizadas por hora en las últimas 24 horas.

---

[120] Una característica tiene "buena utilidad" cuando: 1) Es relevante para el problema que intentamos resolver (por ejemplo, para detectar fraude, el monto y frecuencia de transacciones son útiles, mientras que el color favorito del cliente no lo es), 2) Tiene calidad y completitud aceptable en los datos (pocos valores faltantes o erróneos), 3) Aporta información nueva y no redundante respecto a otras características, 4) Es interpretable y tiene sentido de negocio, y 5) Es computacionalmente eficiente de calcular y mantener. Algunos ejemplos de características útiles derivadas podrían ser: ratios entre variables, promedios móviles, conteos de eventos en ventanas de tiempo, o indicadores de anomalías estadísticas.



*En este gráfico las **transacciones normales** se distribuyen alrededor de un valor medio (100 dólares) con una desviación estándar de 20 dólares. La **transacción anómala** de 300 dólares está marcada con una línea roja discontinua, indicando que se desvía significativamente de las transacciones normales. Este tipo de análisis visual ayuda a identificar posibles actividades fraudulentas al destacar transacciones que no siguen los patrones esperados. Los modelos de IA analizan estos patrones y alertan sobre transacciones inusuales, mejorando la seguridad y protegiendo a los usuarios de fraudes financieros.*

Algunas *características* importantes en el ámbito de la detección de fraude incluyen:

Múltiples Transacciones Pequeñas:

- 🕒 Número de transacciones por hora/día por usuario - *Un incremento súbito en la frecuencia de transacciones puede indicar que una tarjeta robada está siendo probada con montos pequeños antes de realizar fraudes mayores*
- 💰 Suma total de transacciones pequeñas en un período corto - *Los estafadores suelen fragmentar grandes montos en múltiples transacciones pequeñas para evadir los límites de seguridad*
- 📊 Variación en los montos de transacciones consecutivas - *Los patrones muy regulares o sistemáticos en los montos pueden indicar automatización maliciosa*

Velocidad de Transacciones:

- ⚡ Tiempo entre transacciones consecutivas - *Las transacciones demasiado cercanas en tiempo, especialmente si son en diferentes comercios, pueden indicar uso automatizado fraudulento*
- 🌍 Número de transacciones en diferentes ubicaciones geográficas en un corto período - *Es físicamente imposible que un usuario legítimo realice compras presenciales en lugares muy distantes en poco tiempo*

Patrones de Gasto Inusuales:

- 📈 Desviación del gasto promedio histórico del usuario - *Un cambio drástico en el patrón de gasto puede indicar que la cuenta ha sido comprometida*
- 🛍️ Compras en categorías atípicas para el usuario - *Las compras repentinas en categorías nunca antes visitadas por el usuario pueden señalar uso fraudulento*
- ❌ Ratio de transacciones rechazadas vs. aprobadas - *Un alto número de intentos fallidos seguidos de transacciones exitosas puede indicar que se están probando diferentes combinaciones de datos para validar una tarjeta robada*

Comportamiento Geográfico:

- 📍 Distancia entre transacciones consecutivas - *Las distancias físicamente imposibles de recorrer entre compras consecutivas son una señal clara de actividad fraudulenta*
- 🌐 Transacciones en países de alto riesgo - *Ciertas regiones geográficas tienen históricamente mayor incidencia de fraude y merecen mayor escrutinio*
- 🗺️ Cambios repentinos en la ubicación habitual de compra - *Un usuario que siempre compra en una ciudad y repentinamente realiza transacciones en otra región puede indicar compromiso de la cuenta*

Patrones Temporales:

- ⏰ Transacciones fuera del horario habitual del usuario - *Las compras a altas horas de la madrugada cuando el usuario normalmente no opera pueden señalar actividad sospechosa*
- 📅 Frecuencia inusual de transacciones en días festivos o fines de semana - *Los patrones de compra suelen ser consistentes con el estilo de vida del usuario y cambios drásticos pueden indicar fraude*

Comportamiento del Dispositivo:

- 📱 Cambios frecuentes de dispositivo o IP - *El cambio constante de dispositivos puede indicar intentos de evadir la detección de fraude*

- 🔒 Uso de VPN o proxy en transacciones - *El enmascaramiento de la ubicación real puede ser un intento de ocultar actividad fraudulenta*
- 🏠 Discrepancias entre la ubicación del dispositivo y la dirección de facturación - *Grandes diferencias entre estos lugares, especialmente si son persistentes, pueden indicar usurpación de identidad*

Cada una de estas características son las principales generadoras de alertas, porque en el mundo de la actividad fraudulenta y el *lavado de dinero*[121], los criminales aplican estrategias diversas para pasar desapercibidos por los principales métodos de detección. De la misma manera que si alguien roba tu tarjeta de crédito, es más probable que intente realizar múltiples pequeñas transacciones para intentar no ser detectado.

Los sistemas más avanzados de IA en la actualidad incorporan estas características y son cada vez más capaces de detectar actividad fraudulenta, en especial en casos muy complejos que en el pasado eran prácticamente imposible de detectar.

Técnicas de IA en Detección de Fraude

Análisis de Anomalías

Utiliza algoritmos de clustering y Máquinas de Soporte Vectorial para identificar transacciones inusuales.

Aprendizaje Supervisado

Emplea Regresión Logística y Árboles de Decisión para clasificar transacciones como fraudulentas o legítimas.

Redes Neuronales

Aplica Redes Neuronales Artificiales para aprender patrones complejos en los datos financieros.

Paso 3: Análisis de patrones con modelos de IA

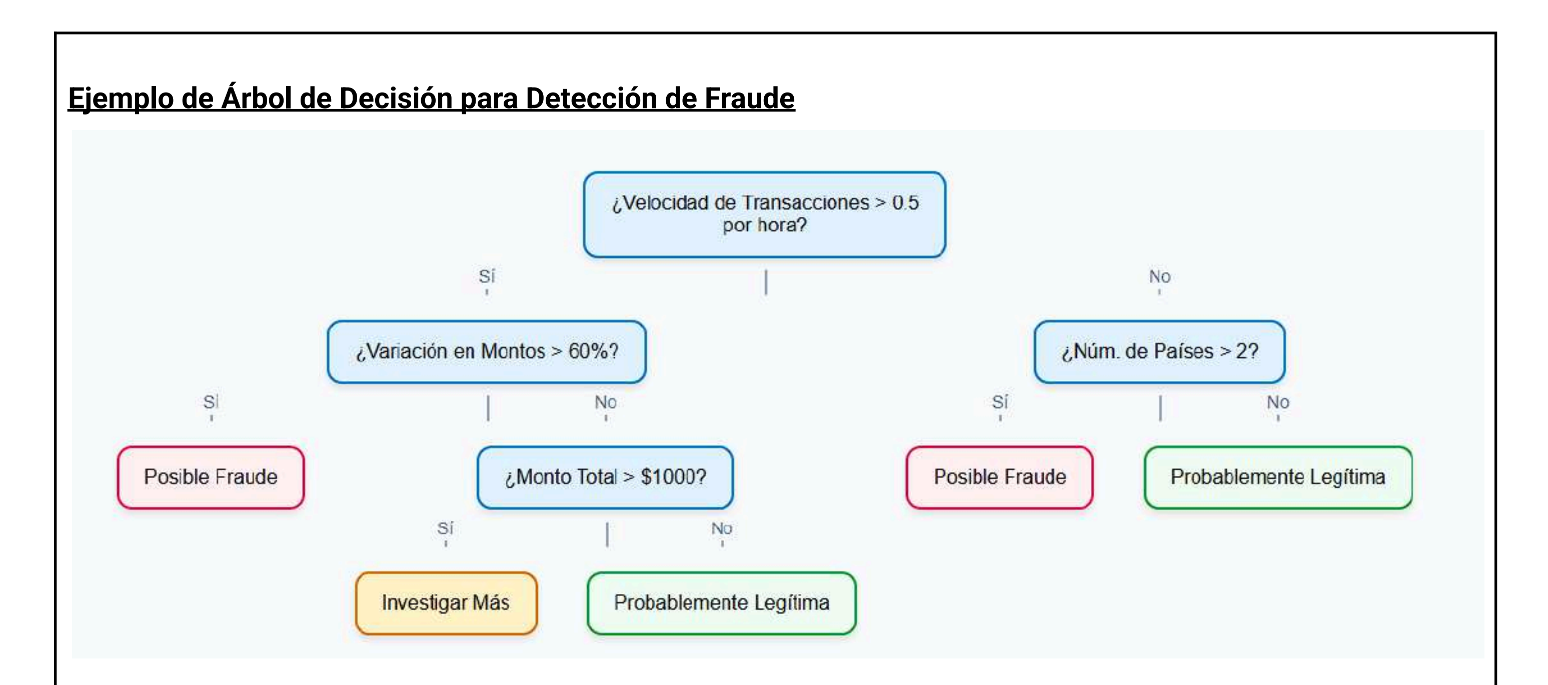
Una vez creadas las *características* necesarias para el caso de uso, se utilizan modelos de IA para analizar estos datos y detectar patrones que indiquen potencial actividad fraudulenta.

[121] El *lavado de dinero* es el proceso mediante el cual se intenta dar apariencia legal a fondos provenientes de actividades ilícitas. Típicamente ocurre en tres etapas: 1) Colocación: introducir el dinero ilícito en el sistema financiero, frecuentemente fragmentando grandes sumas en múltiples transacciones pequeñas ("smurfing") 2) Estratificación: realizar múltiples transacciones complejas para dificultar el rastreo del origen de los fondos, como transferencias entre empresas fantasma o compra-venta de activos 3) Integración: reintroducir el dinero en la economía legal mediante inversiones aparentemente legítimas, como compra de propiedades o negocios. La detección del lavado de dinero es crucial no solo por su vínculo con actividades criminales como narcotráfico, terrorismo y corrupción, sino porque su prevención es una obligación legal para instituciones financieras, que pueden enfrentar severas sanciones por incumplimiento.

Técnicas como el aprendizaje **no supervisado** y el aprendizaje **supervisado** que vimos que pertenecen al *machine learning* son esenciales para este propósito.

Con *machine learning* supervisado, podemos utilizar las *características* que hemos creado para identificar patrones y clasificar transacciones como fraudulentas o legítimas. Veamos un ejemplo simplificado de cómo podría funcionar un árbol de decisión para este propósito.

Ejemplo de Árbol de Decisión para Detección de Fraude



¿Cómo se utilizaría este modelo?

- **Entrada de datos**: El modelo recibe las *características* calculadas para una transacción o conjunto de transacciones de un usuario, es decir, una nueva transacción ha llegado a los sistemas.
- **Evaluación de nodos**: El árbol de decisión evalúa cada característica en orden, comenzando desde la raíz.
- **Toma de decisiones**: En cada nodo, se toma una decisión basada en el valor de la característica, siguiendo la rama correspondiente.
- **Clasificación**: El proceso continúa hasta llegar a una hoja del árbol, que proporciona la clasificación final.

Interpretación de los posibles resultados

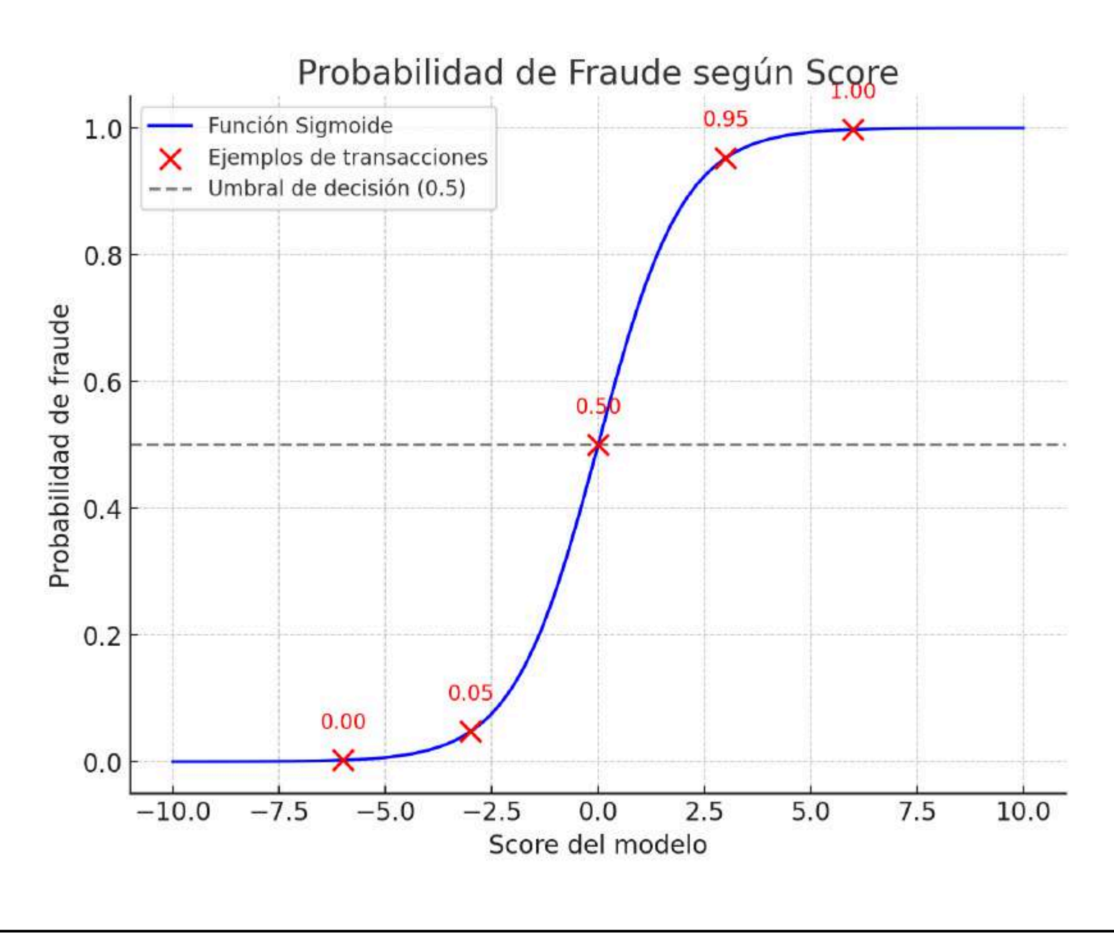
- **Posible fraude**: Estas transacciones se marcan para revisión inmediata o se bloquean automáticamente, dependiendo de las políticas de la empresa.
- **Investigar más**: Estas transacciones requieren una revisión adicional, posiblemente incluyendo verificación con el cliente.
- **Probablemente legítima**: Estas transacciones se consideran de bajo riesgo y generalmente se aprueban automáticamente.

Si en lugar de un árbol de decisión quisiéramos un modelo paramétrico como la regresión logística…

Detección de Fraude: Regresión Logística

Si en lugar de un árbol de decisión quisiéramos un modelo paramétrico como la regresión logística, el siguiente paso en nuestro proceso de detección de fraude sería transformar todas las características que analizamos (transaccionales, geográficas, temporales y de dispositivo) en un modelo matemático capaz de tomar decisiones. Este modelo es especialmente útil cuando necesitamos calcular la probabilidad de que una transacción sea fraudulenta. El proceso comienza con un conjunto de datos históricos donde ya sabemos qué transacciones fueron legítimas y cuáles fraudulentas. Estas transacciones etiquetadas, junto con todas sus características (como los patrones de velocidad de transacciones, variaciones de montos y comportamientos geográficos que analizamos anteriormente), pasan por un proceso de normalización[122] para que todas las variables sean comparables entre sí. El modelo de regresión logística entonces "aprende" qué combinación de estas características es más relevante para identificar fraude, asignando pesos a cada una. El resultado es una herramienta que complementa al árbol de decisión que ya vimos, pero que en lugar de dar una respuesta binaria (fraude/no fraude), nos proporciona una probabilidad que nos permite establecer diferentes niveles de riesgo.

Este modelo produce una curva en forma de S que separa las transacciones en dos categorías: probablemente legítimas y probablemente fraudulentas.



[122] Normalización: Proceso que ajusta las variables a una misma escala (por ejemplo de 0 a 1) para que el modelo pueda compararlas de manera justa. Por ejemplo, si tenemos montos de transacciones que van de $100 a $10,000 y antigüedades de cuenta que van de 1 a 30 días, la normalización los convierte a una escala común donde ambos varían entre 0 y 1, permitiendo que el modelo los compare adecuadamente.

El gráfico de arriba ilustra cómo la probabilidad de que una transacción sea fraudulenta cambia en función del score calculado por el modelo de regresión logística. La curva *sigmoide* transforma el *score* en una probabilidad entre 0 y 1, donde valores altos indican mayor riesgo de fraude. Los puntos rojos representan ejemplos de *scores* específicos y sus probabilidades de fraude correspondientes. La línea discontinua en 0.5 marca el umbral de decisión típico: transacciones con probabilidad superior a 0.5 se consideran posiblemente fraudulentas, mientras que las que están por debajo de este valor se clasifican como legítimas.

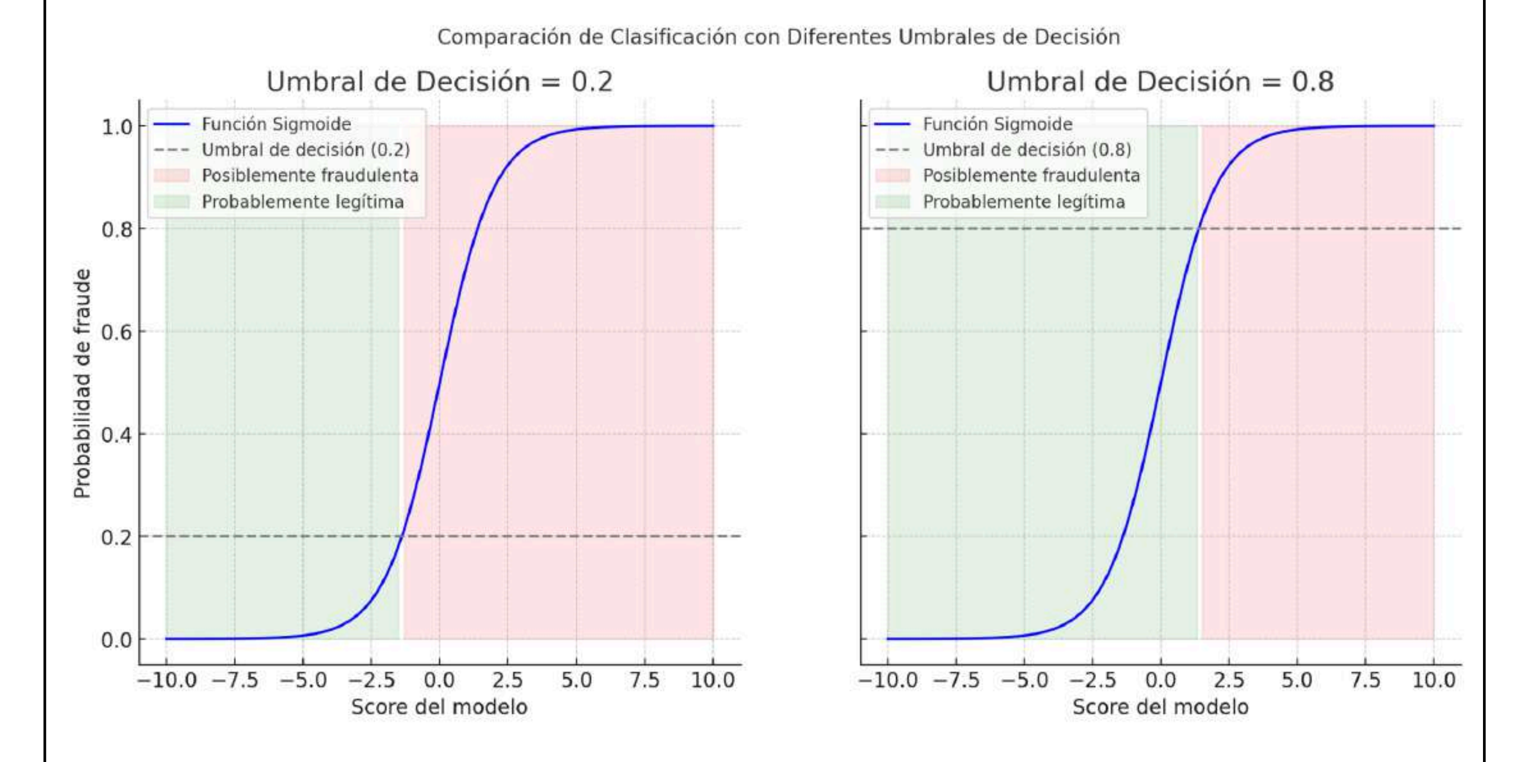
La regresión logística en este contexto funciona de la siguiente manera:

- Se calcula una puntuación de riesgo para cada transacción basada en sus características.
- El modelo de regresión logística toma esta puntuación y calcula la probabilidad de que la transacción sea fraudulenta.
- Si la probabilidad calculada está por encima del umbral de decisión, la transacción se marca como potencialmente fraudulenta para una revisión adicional.
- Las transacciones por debajo del umbral se consideran probablemente legítimas y se procesan normalmente.



El umbral de decisión en un modelo de detección de fraude es ajustable y puede adaptarse al caso de uso específico. Si se establece un umbral bajo, el modelo marcará un mayor número de transacciones como potencialmente fraudulentas, lo cual es útil cuando la prioridad es detectar todos los posibles fraudes, incluso si se generan algunos falsos positivos. Por otro lado, un umbral más alto reduce el número de

alertas, enfocándose solo en transacciones con alta probabilidad de fraude. Este ajuste permite equilibrar entre detectar fraudes y minimizar el número de transacciones revisadas manualmente, optimizando el modelo para objetivos como reducir pérdidas por fraude o controlar costos operativos.



Arriba vemos dos configuraciones de umbral de decisión en un modelo de detección de fraude utilizando regresión logística. A la izquierda, el umbral de decisión es 0.2, lo cual resulta en un área roja más amplia, indicando que más transacciones se consideran potencialmente fraudulentas y serán revisadas. A la derecha, el umbral es 0.8, lo que reduce significativamente el área roja y restringe las alertas a transacciones con una probabilidad de fraude mucho más alta. Esto demuestra cómo un umbral más bajo aumenta la sensibilidad del modelo, mientras que uno más alto se enfoca en transacciones con mayor certeza de fraude, balanceando la cantidad de alertas y la precisión del modelo.

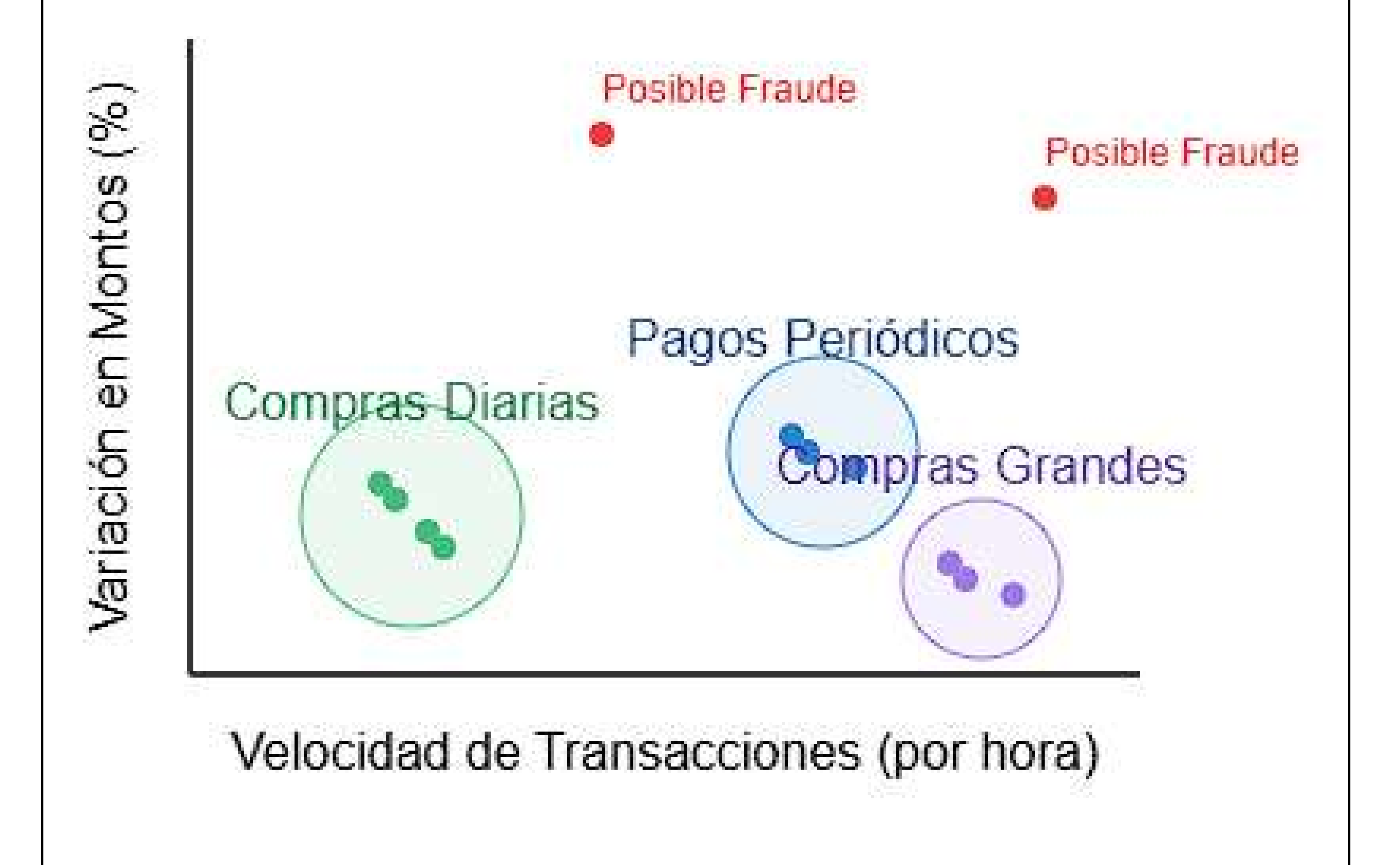
Este enfoque probabilístico de la regresión logística ofrece una ventaja crucial sobre métodos más simples: la capacidad de priorizar y gestionar el riesgo de manera flexible. A diferencia del árbol de decisión que vimos anteriormente, que nos daba decisiones binarias, la regresión logística nos permite establecer diferentes niveles de acción basados en probabilidades. Por ejemplo, podríamos configurar el sistema para que transacciones con probabilidad de fraude superior a 0.8 se bloqueen automáticamente, aquellas entre 0.5 y 0.8 requieran verificación telefónica con el cliente, y las que estén entre 0.2 y 0.5 solo generen una notificación por email. Esta flexibilidad, combinada con la capacidad de ajustar los umbrales según las necesidades específicas del negocio, hace de la regresión logística una herramienta fundamental en los sistemas modernos de detección de fraude, permitiendo a las instituciones financieras balancear eficientemente la seguridad con la experiencia del usuario.

Es importante notar que en el mundo real de la detección de fraude, raramente vamos a tener el escenario ideal donde todas las transacciones históricas están correctamente etiquetadas como "fraude" o "legítimas". Muchos fraudes se descubren tarde o nunca se detectan, y

constantemente aparecen nuevos tipos de estafas que no existen en nuestros datos históricos. Por esto, los sistemas modernos complementan o directamente dependen completamente de los métodos aprendizaje no supervisado, que en lugar de aprender de ejemplos etiquetados, buscan identificar anomalías: transacciones que se desvían significativamente del comportamiento normal. Esta combinación permite detectar tanto fraudes conocidos como nuevos patrones sospechosos.

Detección de Fraude: Clustering con K-means

Como vimos anteriormente, la velocidad de transacciones y la variación en los montos son dos características cruciales para detectar fraude. Por ejemplo, recordemos que un incremento súbito en la frecuencia de transacciones puede indicar que una tarjeta robada está siendo probada, mientras que patrones muy regulares en los montos pueden señalar automatización maliciosa. El algoritmo K-means nos permite analizar estas características sin necesidad de contar con ejemplos etiquetados de transacciones fraudulentas, lo cual es una ventaja significativa cuando enfrentamos nuevos tipos de fraude. Este algoritmo funciona agrupando las transacciones en clusters o grupos basándose en qué tan similares son entre sí: imaginá que cada transacción es un punto en un mapa donde la posición horizontal representa la velocidad de transacciones y la vertical la variación en los montos. K-means va a identificar automáticamente grupos de puntos que están cerca unos de otros, estableciendo patrones de comportamiento "normal". Las transacciones que caen lejos de estos grupos (outliers) son las que nos interesan particularmente, ya que podrían indicar actividad fraudulenta al desviarse significativamente de los patrones habituales de los usuarios.



Interpretación de los Clusters

- **Compras Diarias (Verde)**: Transacciones frecuentes con variación de montos moderada, típico de gastos cotidianos como supermercado o transporte.
- **Pagos Periódicos (Azul)**: Transacciones regulares con montos muy similares, representando suscripciones, servicios o pagos programados.
- **Compras Grandes (Violeta)**: Transacciones menos frecuentes con montos más variables, como compras especiales o pagos esporádicos.

Detección de Anomalías

Los **puntos rojos** representan transacciones que no encajan en ningún patrón normal y podrían indicar:

- Alta frecuencia de transacciones con gran variación en montos (posible prueba de tarjeta robada)
- Comportamiento muy diferente al histórico del usuario
- Patrones que se asemejan a actividades fraudulentas conocidas

La ventaja de usar K-means[123] en este contexto es que puede identificar automáticamente grupos de comportamiento normal sin necesidad de etiquetas previas de fraude. Es importante aclarar que la clave para la detección de fraude no está solo en la identificación de los clusters, sino principalmente en la medición de la distancia entre cada nueva transacción y el centroide del cluster más cercano. Si esta distancia supera significativamente lo normal (por ejemplo, más de 3 desviaciones estándar), la transacción se considera una anomalía potencialmente fraudulenta, incluso si está en una dirección o región del espacio donde no hemos visto fraudes anteriormente. Esto es especialmente útil para detectar nuevos tipos de fraude que aún no han sido identificados por los métodos supervisados que vimos anteriormente.

La detección de fraude es un excelente ejemplo de cómo diferentes técnicas de *machine learning* pueden trabajar en conjunto para crear sistemas más robustos y adaptables. Empezamos viendo cómo crear *features* relevantes a partir de los datos de transacciones, después exploramos métodos supervisados como *árboles de decisión* y *regresión logística* que nos permiten aprovechar el conocimiento de fraudes pasados para entrenar modelos, y finalmente vimos técnicas no supervisadas como *K-means* que nos ayudan a detectar anomalías sin necesidad de ejemplos previos. Esta combinación de enfoques es fundamental porque los defraudadores constantemente desarrollan nuevas tácticas, y nuestros sistemas deben ser capaces tanto de detectar patrones conocidos como de identificar comportamientos sospechosos nunca antes vistos. En un mundo donde las transacciones digitales son cada vez más frecuentes y complejas, la IA se convirtió en una herramienta indispensable para mantener la seguridad financiera de usuarios e instituciones.

[123] Aunque en este ejemplo utilizamos K-means por su simplicidad y porque ya lo estudiamos en detalle anteriormente en el libro, es importante mencionar que en aplicaciones reales de detección de fraude se suelen emplear algoritmos más sofisticados. Por ejemplo, DBSCAN sería más apropiado porque no asume clusters de forma esférica y maneja mejor los outliers, Isolation Forest es particularmente eficiente para detectar anomalías en grandes volúmenes de datos, y Local Outlier Factor (LOF) es especialmente bueno para detectar anomalías basándose en la densidad local de los datos. Sin embargo, K-means nos sirve perfectamente para ilustrar el concepto fundamental de cómo el clustering puede aplicarse a la detección de fraude, ya que los principios básicos (agrupación de comportamientos normales y detección de desviaciones significativas) son similares en todos estos métodos.

05. 📈 Previsión de ventas

La previsión[124] de ventas y rentabilidad suele ser uno de los desafíos más críticos y complejos que enfrentan las empresas modernas. Pensá en una empresa de combustibles que debe decidir cuántos litros de gasolina abastecer a cada una de sus estaciones de servicio del país para atender a la demanda. Suministrar poco combustible significaría correr el riesgo de tener estaciones desabastecidas y clientes que migran a la competencia, mientras que enviar demasiado implicaría tener capital inmovilizado y ocupar capacidad de almacenamiento que podría usarse para otros productos. Esta decisión aparentemente simple se vuelve extremadamente compleja cuando consideramos todos los factores que pueden influir en la demanda:

- 🌞 Patrones estacionales (mayor consumo en vacaciones y fines de semana largos)
- 🎉 Eventos especiales (éxodos masivos en fechas festivas)
- 📈 Tendencias de largo plazo (cambios hacia vehículos eléctricos)
- 🌍 Factores externos (precios internacionales, obras en las rutas, las acciones de la competencia)
- 🦠 Acontecimientos inesperados como una pandemia.

Tradicionalmente, las empresas abordan el problema de la previsión con procesos manuales largos y complejos, donde grandes equipos de analistas y expertos dedican días, semanas o incluso meses a elaborar previsiones (ciclos de *planning-forecasting)*. Estos procesos típicamente involucran reuniones interdepartamentales, hojas de cálculo complejas y ajustes basados en la experiencia de expertos en cada mercado/departamento. Si bien este enfoque funcionó durante décadas, tiene limitaciones importantes: es lento, costoso, difícil de escalar y muy susceptible a sesgos y errores humanos. Las empresas que quieran competir no pueden darse el lujo de seguir destinando este volumen de recursos a un problema que podría ser resuelto casi íntegramente por la IA.

Los modelos de *machine learning* pueden procesar automáticamente cantidades masivas de datos históricos, identificar patrones sutiles que serían imposibles de detectar manualmente y actualizar sus predicciones en tiempo real a medida que llega nueva información. Esto no solo mejora la precisión de las previsiones, sino que también libera a los equipos para que puedan enfocarse en tareas más estratégicas, como la interpretación de resultados y la toma de decisiones basada en las predicciones del modelo.

En mi trabajo actual, esto lo conocemos como AIDI: *AI Decision Intelligence*. El cambio de un paradigma centrado en la utilización de recursos en la previsión a uno centrado en las decisiones. Si volvemos por un momento al ejemplo básico de las recomendaciones de

[124] *Forecasting*, *pronóstico*, *proyección*, son sinónimos.

Netflix, imaginate el tiempo que perderías si tuvieses que recorrer el catálogo entero de películas y series intentando averiguar cuál de ellas ver luego, sin ninguna asistencia de la inteligencia artificial[125]. Desde luego que ese sistema nos cansaría en poco tiempo. En cambio tenemos una recomendación de diferentes alternativas seleccionadas especialmente para nosotros y nuestra única tarea es decidir finalmente qué ver dentro de una pequeña fracción de posibilidades. Esto en esencia es AIDI, son decisiones optimizadas y asistidas por la inteligencia artificial y poco a poco va abarcando cada una de las diferentes decisiones que se pueden tomar en el ámbito empresarial. No se trata de reemplazar personas, no se trata ni siquiera de hacerlo mejor que lo harían las personas, simplemente se trata de que las personas se quiten el trabajo *pesado*, manual y repetitivo, para dedicarse a la toma de decisiones de *valor*.

Previsión de Ventas con IA

La previsión de ventas con inteligencia artificial revoluciona la toma de decisiones empresariales. Permite predecir ventas futuras con precisión, optimizando inventarios y estrategias de marketing.

Esta tecnología utiliza datos históricos y factores externos para generar pronósticos precisos. Explora cómo la IA transforma la planificación comercial.



¿Cómo se realiza la previsión de ventas?

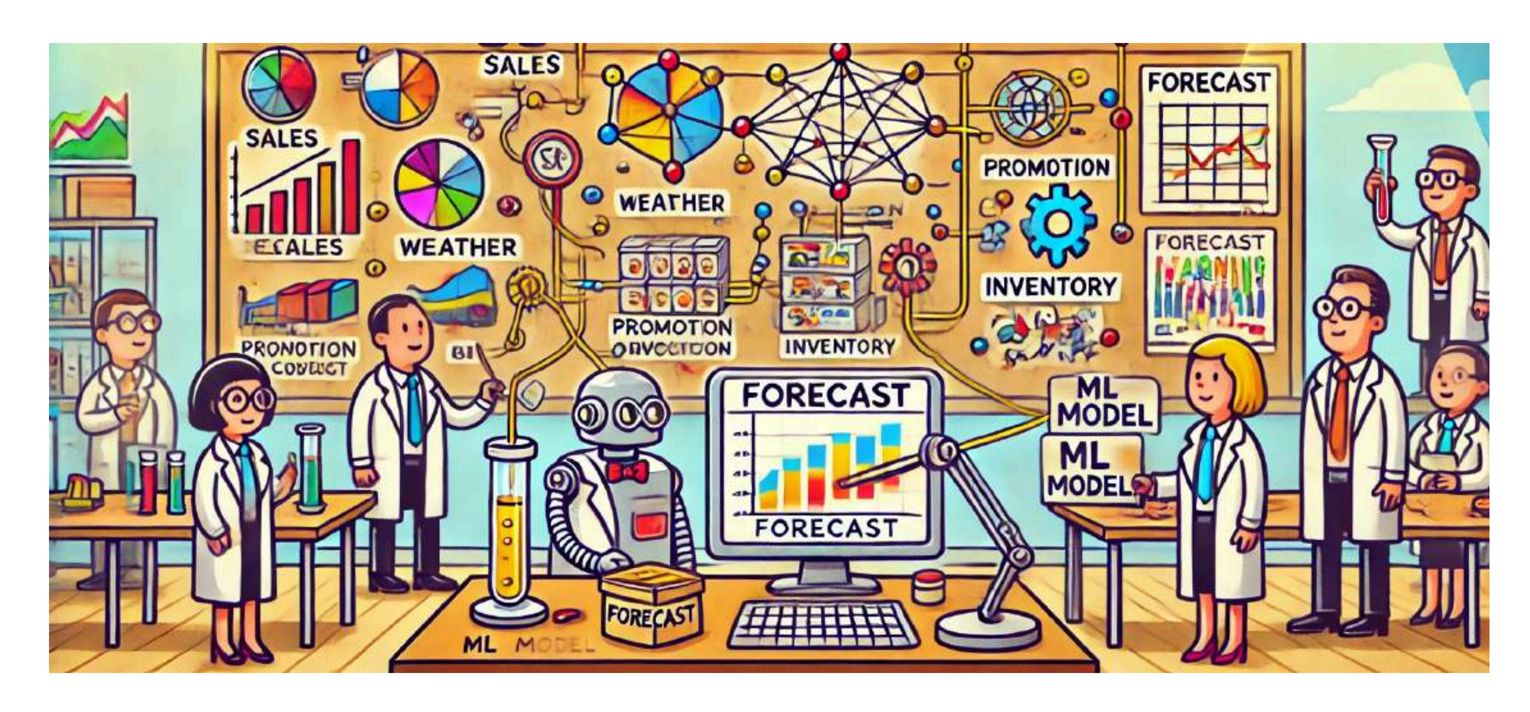
Paso 1: Recopilación de Datos

Primero, obviamente se recopilan datos históricos de ventas. Estos datos pueden incluir:

- Ventas diarias, semanales o mensuales
- Datos de marketing y promociones
- Factores externos como clima, eventos y economía
- Información de inventario

[125] Dejemos de lado por un momento la nostalgia que a algunos nos trae aquellas épocas de ir al videoclub y estar 1 hora decidiendo qué película alquilar :)

● etc.



La siguiente tabla es una muestra de lo que podría ser el resultado de combinar una tabla de ventas diarias histórica con otros datos importantes como si en el día hubo una promoción, un evento, qué tiempo hizo y el nivel de inventarios.

Fecha	Ventas Diarias	Promoción Activa	Evento Externo	Clima	Inventario Disponible
2024-09-01	$5,230	Ninguna	Inicio de mes	Soleado	1200 unidades
2024-09-02	$4,890	Ninguna	Día laboral	Nublado	1150 unidades
2024-09-03	$5,100	Descuento del 10% en electrónicos	Día laboral	Lluvia ligera	1100 unidades
2024-09-04	$5,450	Descuento del 10% en electrónicos	Día laboral	Soleado	1050 unidades
2024-09-05	$6,200	2x1 en productos seleccionados	Víspera de fin de semana	Soleado	980 unidades
2024-09-06	$7,800	2x1 en productos seleccionados	Inicio de fin de semana	Parcialmente nublado	890 unidades
2024-09-07	$8,500	2x1 en productos seleccionados	Fin de semana	Soleado	780 unidades

De la misma forma que vimos en el caso de detección de fraude anterior, la tabla de arriba sería el resultado de transformar los datos de transacciones individuales en un *dataset de características* donde se resume la actividad diaria. Podemos observar características como la suma total de ventas del día, la presencia de alguna promoción, evento externo, el tiempo

que se observó ese día y el inventario disponible. Estos datos sí son útiles para el modelado con *machine learning*[126].



Paso 2: Análisis de datos con modelos predictivos

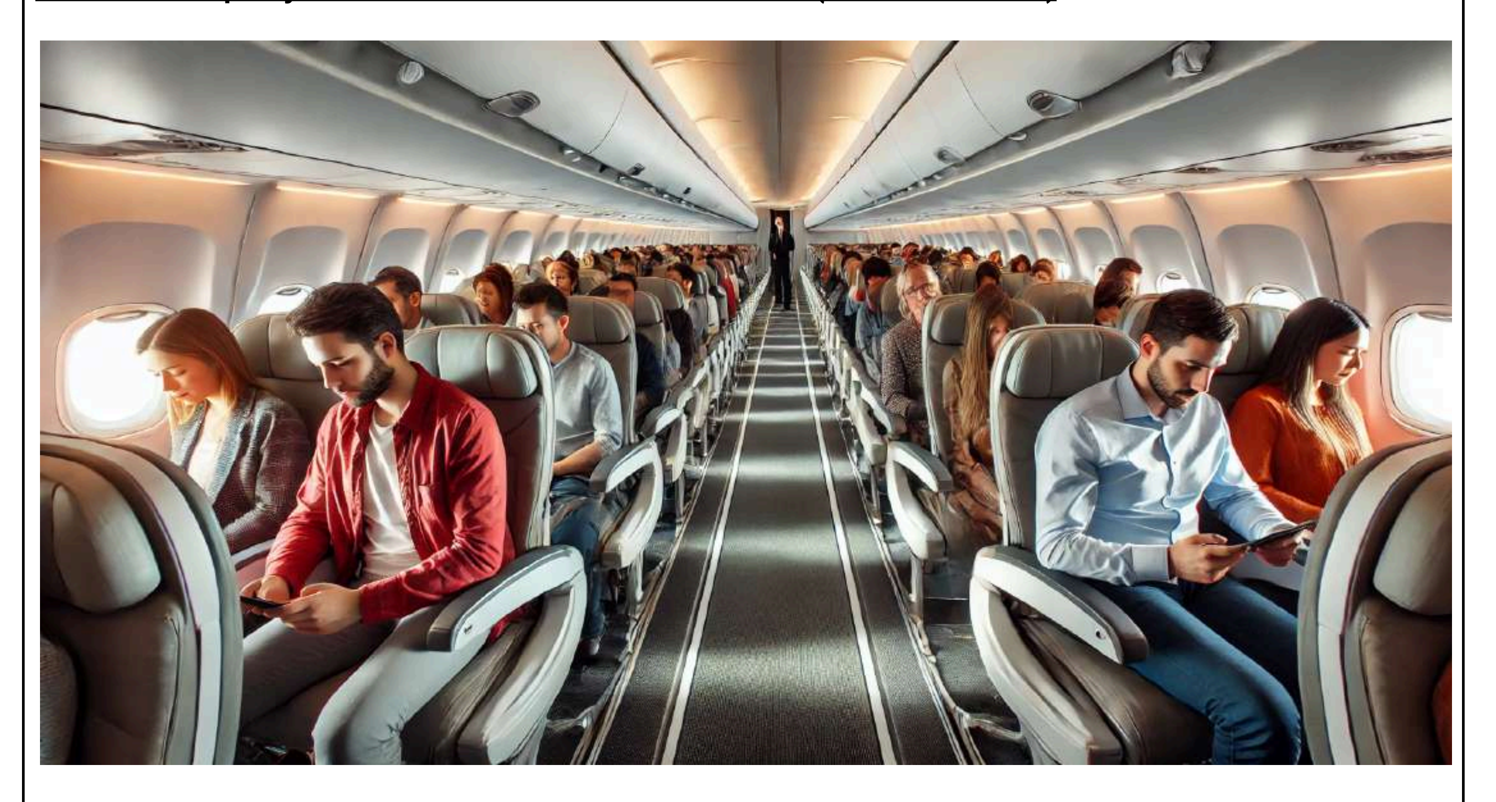
Como ya sabemos, en este paso buscaremos utilizar modelos de IA para analizar estos datos y hacer predicciones sobre las ventas diarias futuras. Técnicas como el análisis de series temporales y el *machine learning* son una vez más esenciales para este propósito.

[126] En sí misma esta tabla necesitaría aún un procesamiento adicional para convertir esa información en variables categóricas o lógicas. Recordá que un modelo de machine learning necesita para funcionar variables numéricas. Por ejemplo la columna de "Evento Externo" podría simplificarse para que sea siempre 0 si no hubo un evento especial, y 1 cuando sí lo hubiera. La columna de "Clima" podría transformarse a (por ejemplo) cinco diferentes columnas donde una de ellas sería "Clima Soleado", siendo nuevamente 0 cuando no fuera soleado y 1 cuando sí (comúnmente conocido como proceso de *one-hot-encoding*).

Técnicas más utilizadas

1. **Análisis de Series Temporales**:
   - **Modelos ARIMA**: Como ya vimos en nuestro caso de uso sobre mantenimiento preventivo, estos modelos son utilizados para predecir valores futuros basados en datos históricos. Vale la pena recordar que ARIMA significa "Autoregressive Integrated Moving Average" (Promedio Móvil Integrado Autorregresivo) y son ampliamente empleados en el análisis de series temporales y la previsión financiera. Los modelos ARIMA son herramientas de predicción que actúan estudiando el pasado para adivinar el futuro. No son modelos de inteligencia artificial sino más bien modelos *estadísticos* que se utilizan hace muchísimos años, pero su efectividad es tal que en el mundo de las series temporales es muy normal que *gane* a la mayoría de modelos de IA por avanzados que sean. A diferencia de los modelos de *machine learning*, los modelos estadísticos como ARIMA solo necesitan de los valores históricos de ventas. Es decir, no utilizan las características adicionales que hemos construído en nuestro *dataset de características* para hacer sus predicciones futuras[127].

Predicción de pasajeros aéreos usando el modelo SARIMA (Seasonal ARIMA)



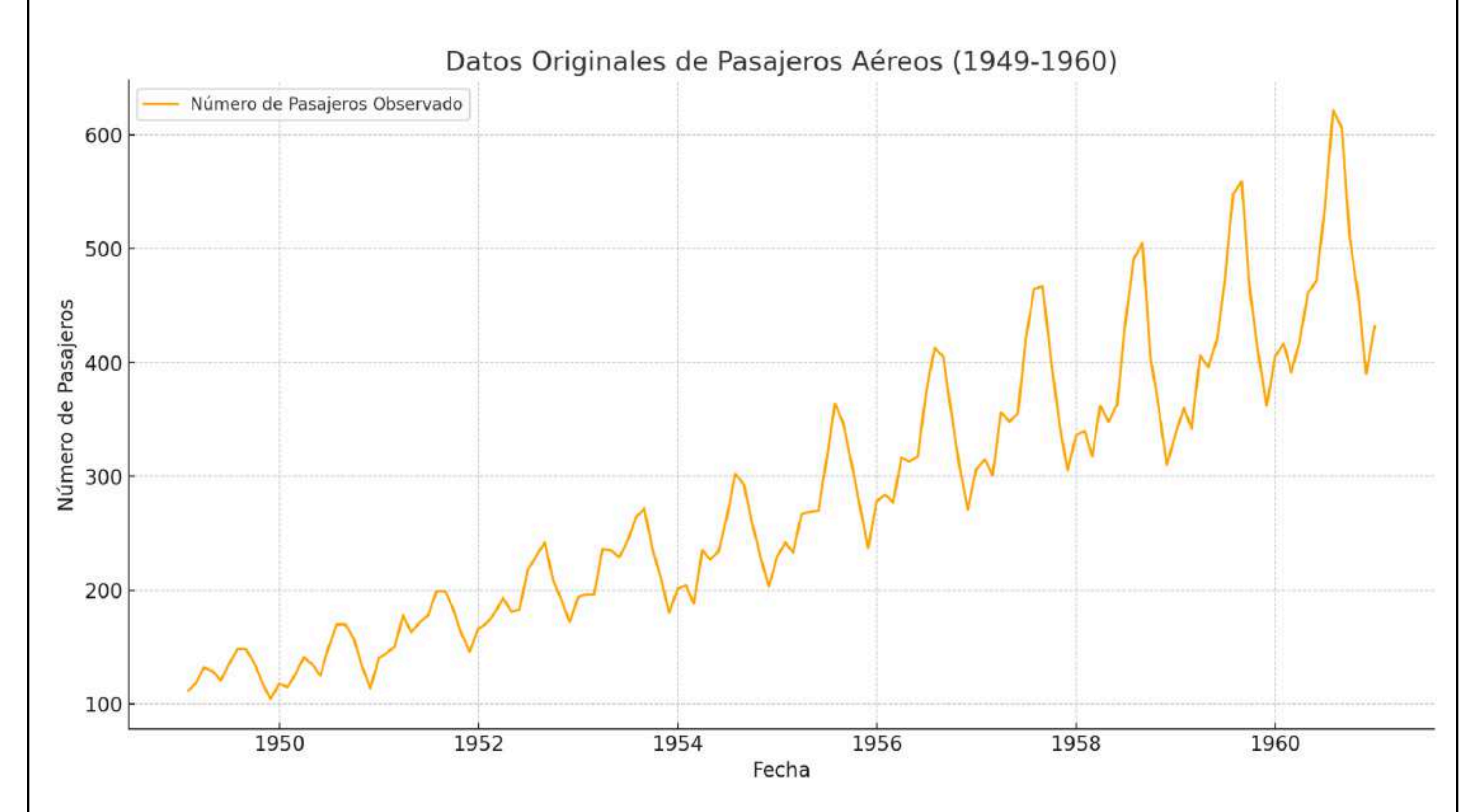
La previsión de la ocupación de pasajeros en las aerolíneas es una práctica esencial para la planificación y

[127] Esto no es del todo cierto, existen variantes de los modelos estadísticos como el ARIMAX que incorpora otros datos exógenos (por eso la X del final) para ayudar al modelo ARIMA base hacer incluso mejores predicciones.

optimización de recursos. En este análisis, utilizamos el modelo SARIMA[128], una extensión del modelo ARIMA que permite manejar patrones estacionales en las series temporales, para prever la ocupación de pasajeros en los próximos tres años.

Visualización de los Datos Originales

Para comenzar, es importante entender el comportamiento histórico de la serie de datos. La siguiente gráfica muestra la evolución mensual del número de pasajeros desde 1949 hasta 1960. Podemos observar un claro patrón de crecimiento a lo largo de los años, acompañado de una estacionalidad anual, donde la demanda de pasajeros varía de manera cíclica.



La **línea amarilla** en el gráfico representa los datos históricos de pasajeros mensuales. Este patrón de estacionalidad es un factor clave que el modelo SARIMA considera para hacer predicciones precisas.

Predicciones con el modelo SARIMA

Una vez que se comprendieron los datos originales, aplicamos un modelo SARIMA para prever la ocupación de los pasajeros en los próximos tres años. Los parámetros elegidos para el modelo SARIMA fueron SARIMA(2, 1, 2)(1, 1, 1, 12)[129]. Estos se seleccionaron para capturar adecuadamente la estructura de la serie temporal, que incluye tanto componentes de tendencia como estacionales con periodicidad anual. En la práctica, la determinación de estos parámetros se realiza observando los gráficos de **autocorrelación (ACF)** y **autocorrelación parcial (PACF)**, los cuales ayudan a identificar los valores apropiados de p, d, q para la parte no estacional y P, D, Q, s para la parte estacional. Además, se prueban diferentes combinaciones de

[128] Seasonal ARIMA = SARIMA

[129] SARIMA(2, 1, 2)(1, 1, 1, 12) representa la configuración del modelo, donde los primeros tres números (2, 1, 2) corresponden a los parámetros del modelo no estacional: p (autorregresivo), d (diferenciación), y q (media móvil). Los siguientes cuatro números (1, 1, 1, 12) se refieren a los componentes estacionales: P (autorregresivo estacional), D (diferenciación estacional), Q (media móvil estacional) y s (periodicidad estacional), en este caso 12, que refleja la estacionalidad anual en datos mensuales.

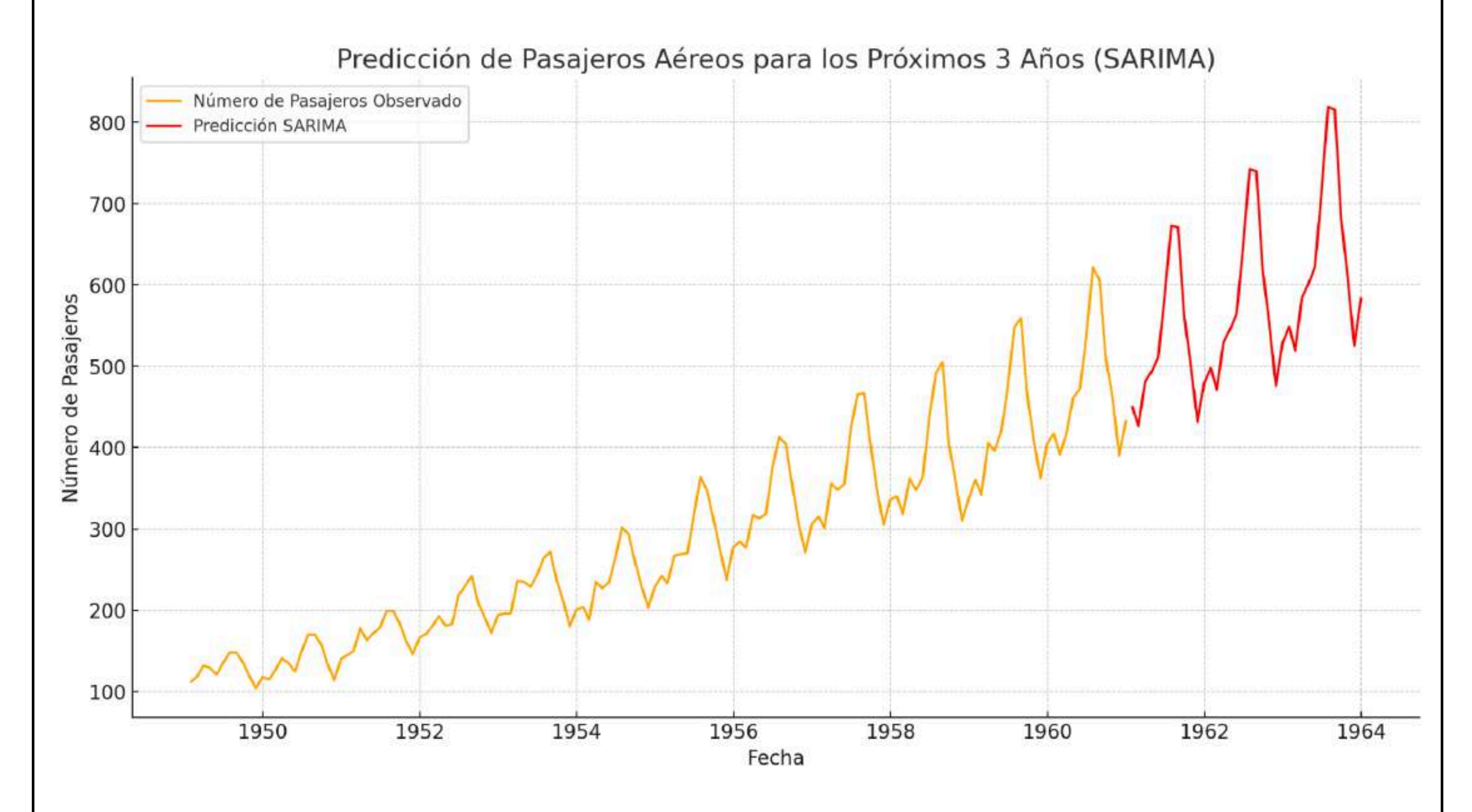
estos parámetros y se evalúan los modelos con criterios como el **Akaike Information Criterion (AIC)** para encontrar el que ofrezca el mejor equilibrio entre ajuste y complejidad del modelo.

Para entender esto de manera más simple, podemos pensar en SARIMA como una herramienta que analiza patrones en los datos mensuales que se repiten tanto en el corto como en el largo plazo. Por ejemplo, en el caso de los pasajeros, el modelo puede identificar patrones que se repiten mes a mes (como tendencias graduales) y anualmente (como los aumentos en temporada de vacaciones). Los números en el modelo (2, 1, 2)(1, 1, 1, 12) son como instrucciones que le dicen al modelo exactamente cuántos meses atrás debe mirar para encontrar estos patrones y cómo debe usarlos para hacer predicciones futuras.

Por supuesto que antes de aplicar el modelo SARIMA se realizó un preprocesamiento básico de los datos. Primero, se aplicó una **transformación logarítmica** a la serie original para estabilizar la varianza y reducir la influencia de los valores extremos. Esta transformación es útil cuando los datos muestran un crecimiento no lineal, ya que ayuda a resaltar los patrones estacionales y la tendencia subyacente[130]. Una vez transformada, la serie fue utilizada directamente en el modelo SARIMA para capturar tanto la estacionalidad como la tendencia en las predicciones.

La siguiente gráfica muestra los datos históricos junto con las predicciones del modelo:

- La **línea roja** en la gráfica indica las predicciones del modelo SARIMA para los próximos tres años.
- La **línea amarilla** representa los datos históricos de la serie temporal.



Como se observa en el gráfico, el modelo SARIMA logra capturar la estacionalidad anual y la tendencia de la serie, proporcionando una previsión coherente que proyecta los patrones observados en el pasado hacia el

[130] Imaginá que tenés números que crecen cada vez más rápido, como 100, 200, 400, 800, 1600. Al aplicar logaritmos, estos números se convierten en una secuencia más uniforme (4.6, 5.3, 6.0, 6.7, 7.4), lo que hace más fácil para el modelo detectar patrones y realizar predicciones precisas. Es como "suavizar" los datos para que los patrones sean más claros y manejables.

futuro.

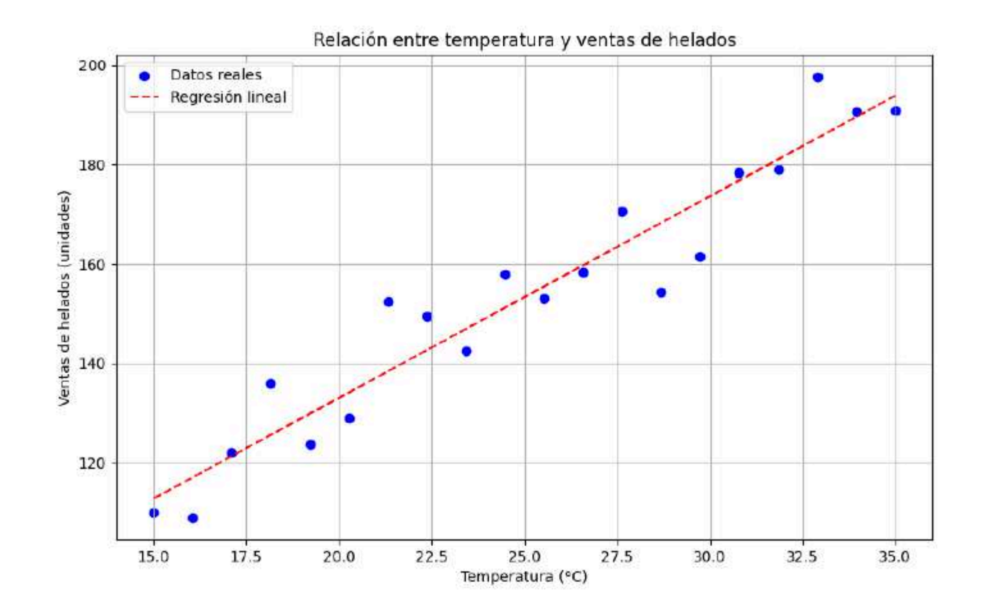
En la práctica, además de realizar predicciones puntuales, es importante calcular el **intervalo de confianza** de las previsiones. Este intervalo proporciona un rango dentro del cual es probable que se encuentren los valores futuros, con un nivel de confianza específico (por ejemplo, 95%). Los intervalos de confianza permiten a los analistas y tomadores de decisiones evaluar la incertidumbre asociada a las predicciones, ayudando a gestionar los riesgos y a planificar con más seguridad. Incluir estos intervalos es esencial para tener una visión más completa y realista de las posibles variaciones en los resultados futuros.

De todo lo que vimos en este ejemplo, me interesa más que nada que te acuerdes que los modelos ARIMA y SARIMA son **autorregresivos**, lo que significa que utilizan los valores pasados de la propia serie temporal para predecir los futuros. En esencia, el modelo analiza cómo los datos históricos de pasajeros se comportan a lo largo del tiempo, identificando patrones y relaciones en la secuencia de valores. Al incorporar tanto los componentes de tendencia como de estacionalidad, el modelo es capaz de proyectar esos patrones hacia el futuro. Este proceso se denomina *autorregresión* porque el modelo se "retroalimenta" de los datos anteriores para generar sus predicciones, basando sus estimaciones en la dependencia entre los valores pasados y los futuros.

Por otro lado, y como veremos a continuación, los modelos de *machine learning* se diferencian del enfoque autorregresivo, como el modelo SARIMA, en que suelen considerar múltiples *características* (*features*) para realizar sus predicciones, no solo los valores pasados de la serie temporal. Mientras que un modelo *autorregresivo* se basa exclusivamente en la *autocorrelación* de los datos históricos para prever el futuro, los modelos de *machine learning* pueden integrar información adicional, como variables externas relacionadas con la economía, el clima, o eventos especiales, lo que les permite capturar patrones más complejos. Estos modelos, como los basados en árboles de decisión, redes neuronales y otros algoritmos avanzados que ya vimos en reiteradas oportunidades, *aprenden* de los datos a través de procesos de entrenamiento-optimización. Esto los hace más flexibles y capaces de adaptarse a cambios en las condiciones que un modelo puramente *autorregresivo* podría no captar.

2. ***Machine Learning* con aprendizaje supervisado**:
    - **Regresión lineal**: a pesar de su aparente simplicidad, sigue siendo uno de los mejores modelos usados en la previsión de ventas. Su verdadero poder está en su capacidad para establecer relaciones claras y directas entre múltiples factores y el resultado que queremos predecir. Imaginemos por un momento una heladería. El modelo de regresión lineal podría analizar factores como la temperatura del día, si es fin de semana, si hay algún evento especial en la zona, y el histórico de ventas previas. Cada uno de estos factores recibe un "peso" o importancia en la predicción final. Por ejemplo, el modelo podría descubrir que por cada grado que sube la temperatura, las ventas aumentan en promedio 5 unidades, o que los fines de semana las ventas son típicamente un 40% superiores a los días entre semana. Lo bueno de la regresión lineal es su *interpretabilidad*. A diferencia de modelos más complejos que funcionan como "cajas negras", la regresión lineal nos permite entender exactamente cómo cada variable afecta a nuestras predicciones. Esto es muy valioso para los dueños del negocio, que necesitan no solo predicciones precisas, sino también entender el

"por qué" detrás de las predicciones. Sin embargo, es importante reconocer sus limitaciones.



La regresión lineal, como su nombre indica, asume relaciones lineales entre variables. En el mundo real, estas relaciones son por lo general más complejas. Por ejemplo, la relación entre la temperatura y las ventas de helados podría no ser lineal: el aumento en las ventas podría ser mucho más pronunciado entre 25°C y 30°C que entre 15°C y 20°C.

- **Árboles de decisión y Bosques Aleatorios (*random forest*)**: los árboles de decisión y, especialmente, los bosques aleatorios, representan un salto cualitativo en nuestra capacidad para capturar relaciones más complejas en los datos. Como ya hemos visto varias veces en este libro, estos modelos funcionan de una manera que podríamos considerar más "humana": tomando decisiones secuenciales basadas en diferentes criterios. Un árbol de decisión para predicción de ventas podría comenzar preguntando: "¿Es la temperatura superior a 25°C?". Si la respuesta es sí, podría seguir con "¿Es fin de semana?", y así sucesivamente. Cada respuesta nos lleva por un camino diferente en el árbol hasta llegar a una predicción final. Esta estructura sí que permitiría capturar interacciones no lineales entre variables de una manera que la regresión lineal no puede.



Los bosques aleatorios (*random forest*) llevan este concepto un paso más allá al combinar cientos o miles de árboles de decisión, cada uno ligeramente diferente. Es como tener un comité de expertos, donde cada experto mira los datos desde una perspectiva ligeramente diferente. La predicción final es un promedio de todas estas opiniones individuales, lo que típicamente resulta en predicciones más robustas y precisas que las de un único árbol. Una característica particularmente valiosa de los bosques aleatorios es su capacidad para manejar diferentes tipos de variables sin necesidad de transformaciones complejas.

Pueden trabajar simultáneamente con variables numéricas (como temperatura o precio) y categóricas (como día de la semana o tipo de promoción) de manera natural.

- **Redes Neuronales Artificiales (ANN)**: las redes neuronales representan el paradigma de la sofisticación en modelos de predicción de ventas. Inspiradas en el funcionamiento del cerebro humano como ya hemos visto, estas redes pueden detectar patrones extremadamente complejos en los datos, patrones que podrían ser invisibles para modelos más simples. En el contexto de la predicción de ventas, una red neuronal puede procesar al mismo tiempo una cantidad enorme de información: datos históricos de ventas, información meteorológica, datos económicos, tendencias en redes sociales, y mucho más. La red aprende automáticamente qué combinaciones de estos factores son más relevantes para predecir las ventas futuras.



La verdadera *magia* de las redes neuronales radica en su capacidad para aprender representaciones jerárquicas de los datos. En las primeras capas de la red, podrían identificarse patrones básicos (como la relación entre temperatura y ventas), mientras que en capas más profundas podrían detectarse patrones más sutiles y complejos (como la interacción entre múltiples variables que afectan a las ventas). Sin embargo, este poder viene con un costo. Las redes neuronales

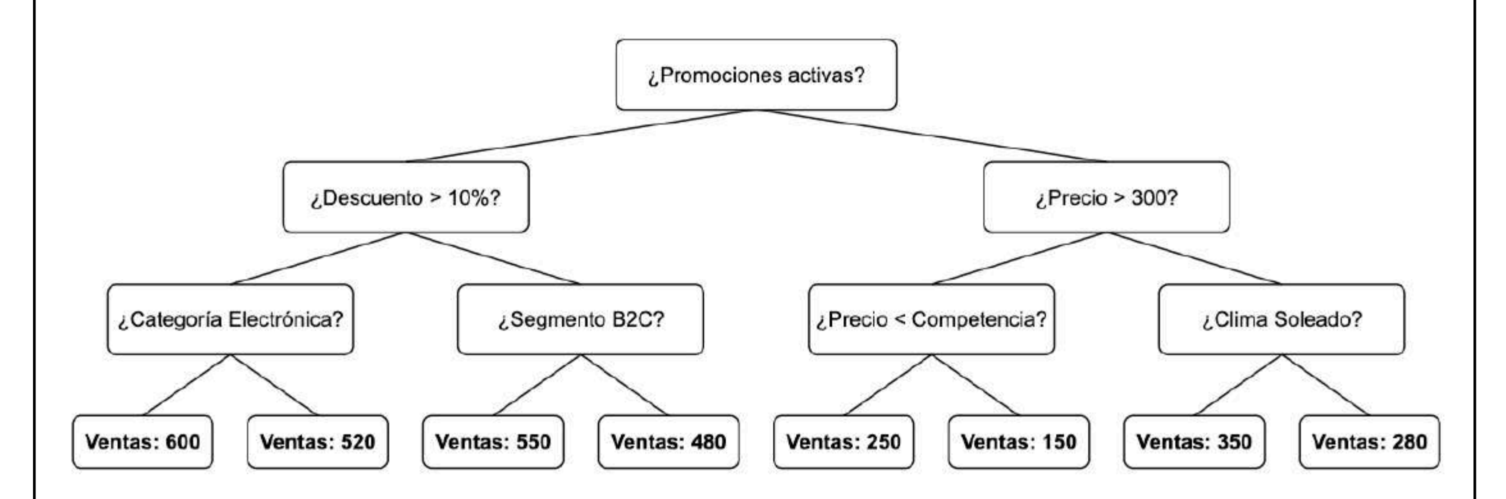
requieren típicamente grandes cantidades de datos para entrenar efectivamente, y su proceso de entrenamiento puede ser computacionalmente intensivo. Además, son notoriamente difíciles de interpretar - a menudo funcionan como una "caja negra" donde es difícil entender exactamente cómo llegaron a una predicción específica.

Ejemplo con *machine learning*

Imaginemos que tenemos un dataset como el de abajo, donde hemos recopilado una serie de *features* o *características* asociadas a las ventas de un comercio. Un dataset con múltiples características como precios, promociones, inventario, clima y competencia podría permitir a los modelos de *machine learning* capturar relaciones complejas y no lineales entre las variables, algo que los modelos estadísticos como ARIMA no logran con la misma eficacia. Mientras que los modelos estadísticos más tradicionales se enfocan en patrones temporales lineales y tienen limitaciones para manejar múltiples variables externas, los modelos de *machine learning* pueden integrar diversos factores, adaptarse a cambios dinámicos en el mercado, y escalar con grandes volúmenes de datos, ofreciendo así predicciones de ventas más precisas y personalizadas.

Fecha	ID_Producto	Categoría_Producto	ID_Tienda	Región	Segmento_Cliente	Precio	Descuento	Promociones	Precio_Competencia	Nivel_Inventario	Gasto_Marketing	Día_Semana	Feriado	Clima	Ventas
2024-09-01	101	Electrónica	1	Norte	B2C	499	10	1	475	100	5000	Lunes	0	Claro	300
2024-09-02	101	Electrónica	2	Sur	B2C	499	0	0	480	80	4500	Martes	0	Lluvia	220
2024-09-03	102	Electrodomésticos	1	Norte	B2B	250	5	1	245	150	7000	Miércoles	0	Nublado	180
2024-09-04	103	Ropa	3	Este	B2C	75	20	1	70	50	2000	Jueves	1	Soleado	500
2024-09-05	104	Electrónica	2	Sur	B2C	999	15	0	950	200	6000	Viernes	0	Claro	400

Un **árbol de decisión** que aprenda de estas características y pueda hacer previsión de ventas podría lucir así:



Acá tenés algunos ejemplos de cómo interpretar este árbol de decisión (tené en cuenta que siempre la opción de la izquierda es "Si" y la de la derecha es "No":

1. 🔌💸 **Escenario de alto descuento en electrónica**: Si hay promociones activas, el descuento es mayor al 10%, y el producto es de la categoría Electrónica, las ventas serían de 600 unidades.
2. 💰☀️ **Escenario de precio alto con buen clima**: Si hay promociones activas, el precio es mayor a 300, no es más barato que la competencia, pero el clima está soleado, las ventas serían de 350 unidades.
3. 🛒🏷️ **Escenario de segmento B2C (*business to consumer*) con descuento moderado**: Si hay promociones activas, el descuento es mayor al 10%, pero el producto no es de electrónica sino del segmento B2C, las ventas serían de 550 unidades.
4. 💼📉 **Escenario de precio alto pero competitivo**: Si hay promociones activas, el precio es mayor a 300, pero es más bajo que la competencia, las ventas serían de 250 unidades.
5. 🤷‍♂️🔍 **Escenario sin grandes descuentos ni precios altos**: Si hay promociones activas, pero el descuento no supera el 10% y el precio no es mayor a 300, el árbol no proporciona información sobre las ventas en este caso.

Veamos ahora cómo se abordaría el mismo problema utilizando un modelo de regresión lineal. A diferencia del árbol de decisión, que divide los datos en grupos basados en decisiones binarias, la regresión lineal busca encontrar una relación matemática directa entre las variables predictoras y las ventas.

Para nuestro ejemplo, consideremos las mismas variables que usamos en el árbol de decisión. La ecuación de regresión lineal podría verse así:

$$\text{Ventas} = \beta_0 + \beta_1(\text{Descuento}) + \beta_2(\text{Precio}) + \beta_3(\text{EsElectrónica}) + \beta_4(\text{EsB2C}) + \beta_5(\text{ClimaSoleado}) + \beta_6(\text{MásBaratoQueCompetencia}) + \varepsilon$$

Donde:

- $\boldsymbol{\beta_0}$ es la intercepción (ventas base cuando todas las variables son 0)
- $\boldsymbol{\beta_1}$ a $\boldsymbol{\beta_6}$ son los coeficientes que el modelo aprende de los datos
- $\boldsymbol{\varepsilon}$ es el término de error

Supongamos que después de entrenar el modelo con nuestros datos históricos, obtenemos los siguientes coeficientes:

- **β_0 (Intercepción)** = 200 unidades
- **β_1 (Descuento)** = 15 unidades por cada 1% de descuento
- **β_2 (Precio)** = -0.5 unidades por cada $1 de precio
- **β_3 (EsElectrónica)** = 100 unidades si es producto electrónico
- **β_4 (EsB2C)** = 80 unidades si es producto B2C
- **β_5 (ClimaSoleado)** = 50 unidades si hay sol
- **β_6 (MásBaratoQueCompetencia)** = 70 unidades si somos más baratos

Analicemos los mismos escenarios que vimos en el árbol de decisión:

1. 🔌💸 **Escenario de alto descuento en electrónica**:
    - Descuento del 15%: 15 × 15 = 225 unidades
    - Es electrónica: +100 unidades
    - Precio base: 200 unidades
    - Total ≈ 525 unidades
2. 💰☀️ **Escenario de precio alto con buen clima**:
    - Precio $350: -0.5 × 350 = -175 unidades
    - Clima soleado: +50 unidades
    - Precio base: 200 unidades
    - Total ≈ 75 unidades
3. 🛒🏷️ **Escenario B2C con descuento moderado**:
    - Descuento del 12%: 15 × 12 = 180 unidades
    - Es B2C: +80 unidades
    - Precio base: 200 unidades
    - Total ≈ 460 unidades

¿Cuáles serían las diferencias clave con el árbol de decisión?:

1. **Linealidad**:
    - En la regresión lineal, el efecto de cada variable es constante y aditivo. Por ejemplo, un 1% más de descuento siempre suma 15 unidades a las ventas, independientemente de otras variables.
    - En el árbol de decisión, el efecto puede variar según el contexto y las combinaciones de otras variables.
2. **Interpretación**:
    - La regresión lineal nos proporciona una interpretación precisa y cuantificable de cada variable: "Por cada punto porcentual de descuento, las ventas aumentan en 15 unidades", "Por cada dólar que aumenta el precio, las ventas disminuyen en 0.5 unidades". Esto facilita entender el impacto individual de cada variable y hacer estimaciones rápidas mentalmente.
    - El árbol de decisión, en cambio, nos da reglas de decisión más específicas pero menos flexibles: "Si el descuento es mayor al 10% Y es electrónica, entonces las ventas serán 600". Estas reglas son más fáciles de seguir como proceso de decisión, pero no nos dicen qué sucede en situaciones intermedias (por ejemplo, con un 9.5% de descuento).

3. **Predicciones**:
    - La regresión lineal puede generar predicciones continuas en cualquier punto del espacio de características. Por ejemplo, puede predecir ventas para un descuento de 7.3% o 12.8%, interpolando suavemente entre los valores. Esto la hace más flexible pero puede ser menos precisa en casos extremos donde la relación no es realmente lineal.
    - El árbol de decisión sólo puede predecir los valores específicos que aparecen en sus hojas, basados en los datos de entrenamiento. En nuestro ejemplo, si todas las predicciones son 600, 350, 550 o 250 unidades, no puede predecir un valor intermedio como 425 unidades. Esto lo hace más rígido pero potencialmente más preciso cuando los datos realmente siguen patrones discretos.
4. **Interacciones**:
    - En este modelo de regresión lineal simple, no estamos capturando interacciones entre variables
    - El árbol de decisión captura naturalmente estas interacciones a través de su estructura jerárquica

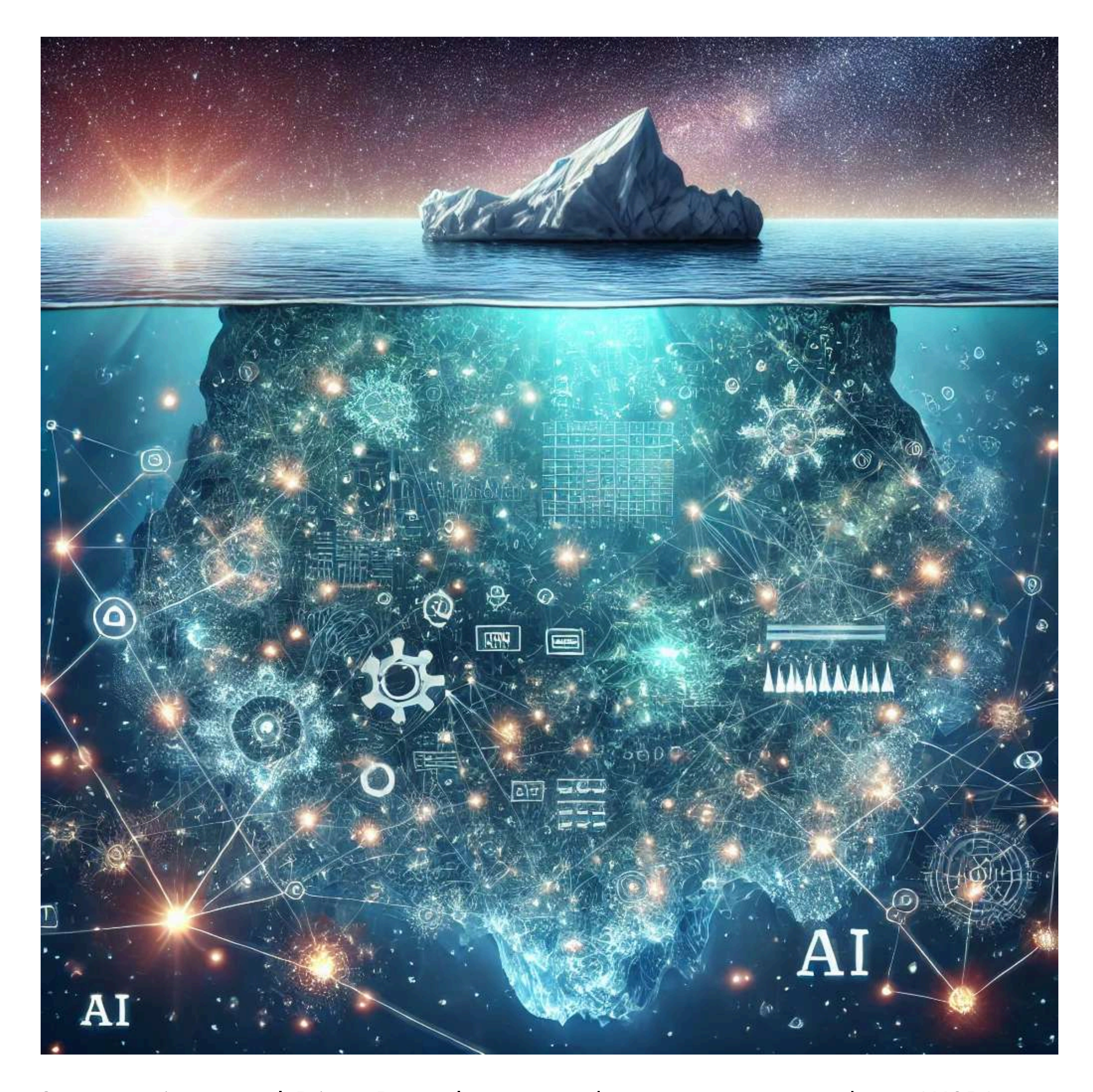
    Por ejemplo, podríamos mejorar nuestro modelo de regresión lineal añadiendo términos de interacción:

    Ventas = β_0 + β_1(Descuento) + β_2(Precio) + β_3(EsElectrónica) + **β_4(Descuento×EsElectrónica)** + ...

    Esto permitiría que el efecto del descuento sea diferente para productos electrónicos versus no electrónicos, acercándose más a la flexibilidad del árbol de decisión, pero manteniendo la interpretabilidad característica de la regresión lineal.

Es importante notar que en la práctica, rara vez encontraremos relaciones tan perfectamente lineales. Por eso, aunque la regresión lineal es una herramienta poderosa y útil como punto de partida, a menudo necesitamos modelos más sofisticados (como los árboles de decisión o las redes neuronales) para capturar la complejidad total de los patrones de ventas en el mundo real.

¡Y así es como terminamos nuestro recorrido por algunos de los casos más fascinantes de la IA en acción! Como podés ver, la inteligencia artificial está haciendo cosas increíbles, desde la híper personalizar contenido y ofertas, la detección de fraudes que antes eran imposibles, la generación de lenguaje humano perfecto y hasta se anima a predecir el futuro de los negocios. Lo más emocionante es que esto es solo la punta del iceberg, hay muchísimas más aplicaciones ahí afuera y muchísimas más esperando a ser descubiertas y desarrolladas.

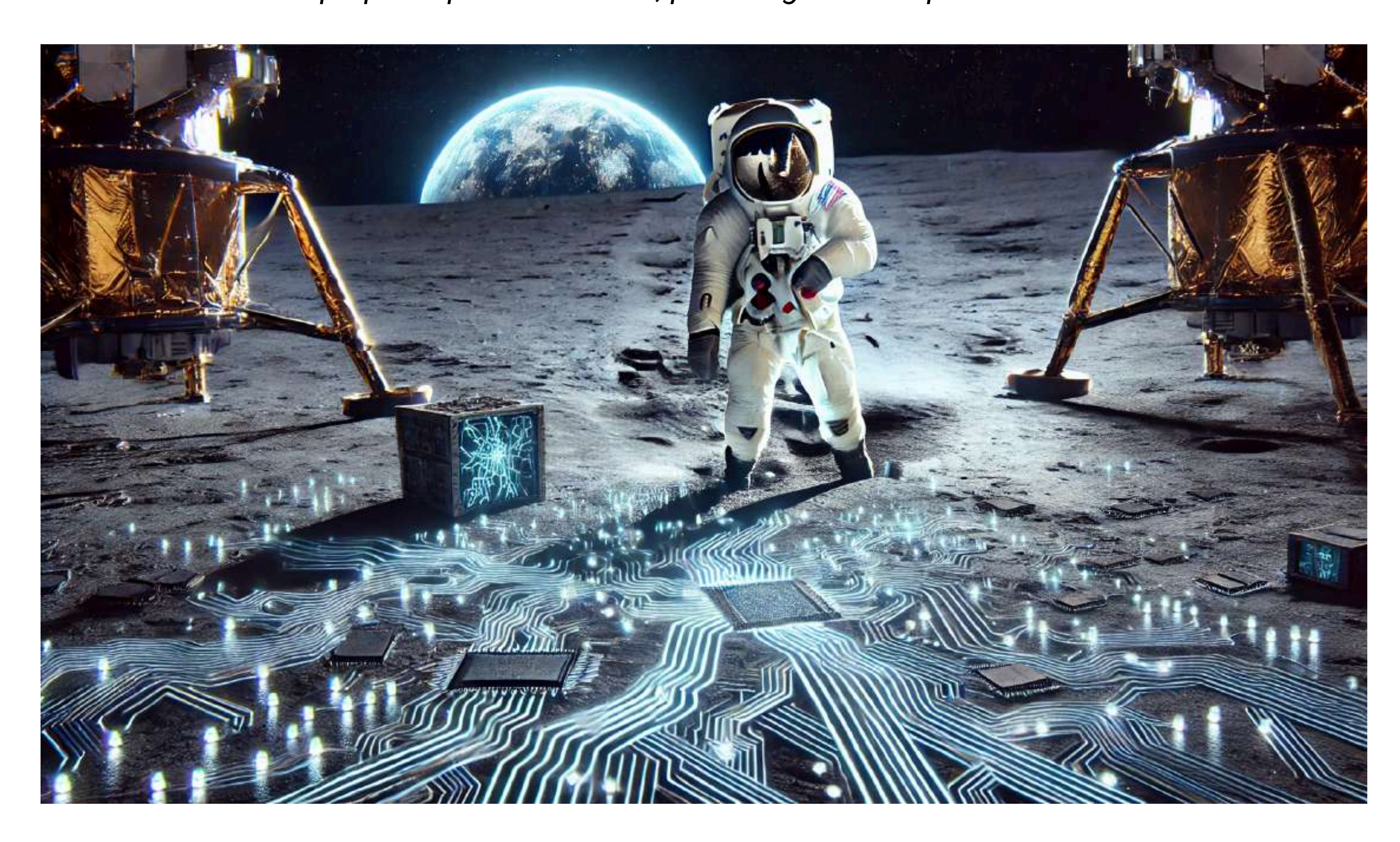


Como ya vimos en el *Primer Paso*, el momento de entrar en este mundo es AHORA, y tu *Segundo Paso* fundamental, el de entender los básicos de la IA lo acabás de completar! No solo eso! Además ahora tenés una mejor idea de los fundamentos matemáticos y estadísticos que hay detrás de esos conceptos y por sobre todas las cosas, pudiste ver casos de uso reales de cómo se ponen en práctica. Considero que de ahora en más estás en perfectas condiciones de poder interpretar y tener conversaciones acerca de mucha de la información que se comunica diariamente acerca del presente y del futuro de la inteligencia artificial y tratar de razonarlo bajo la luz de alguno de todos esos temas que han sido parte de

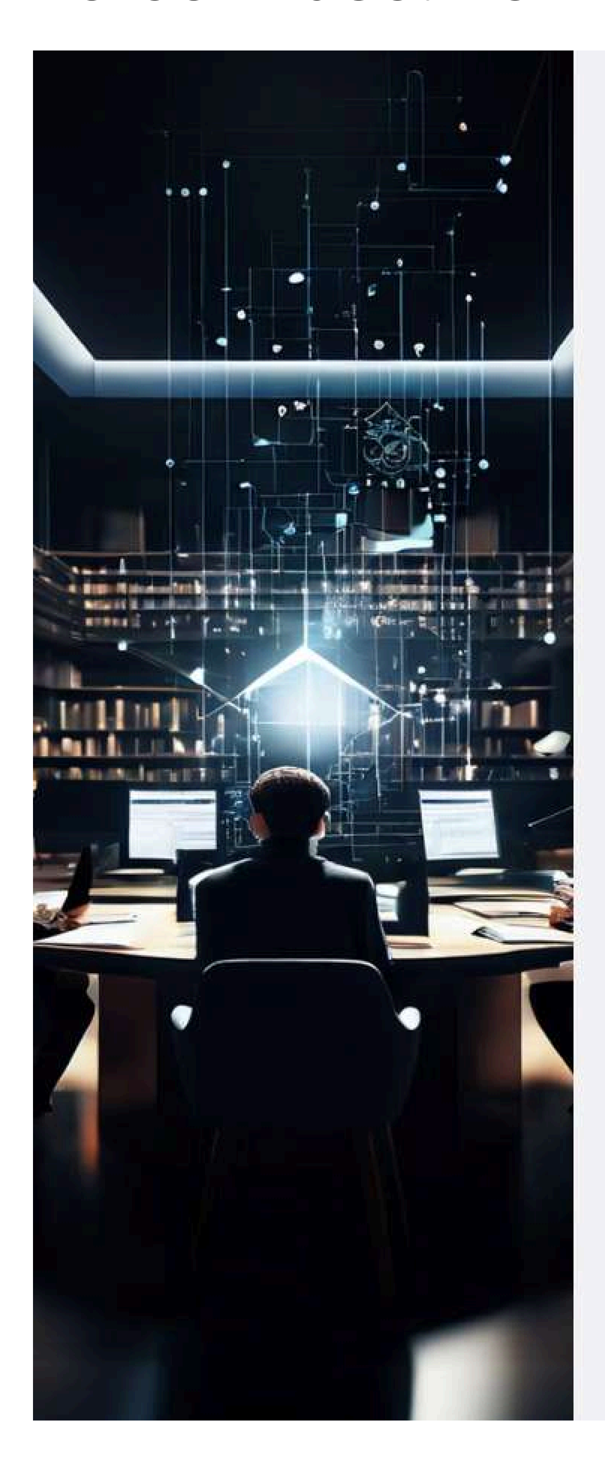
tu *Segundo Paso*. No te va a ser fácil, no te va a ser inmediato, pero tenés las herramientas a disposición para hacerlo. No tengo dudas. No obstante…

…para poder ser realmente parte de esta revolución tecnológica, es importante que entiendas que necesitás más que solo estos conceptos básicos que vimos hasta ahora. Necesitás comprometerte a seguir aprendiendo y creciendo en este campo que cambia día a día. Por eso, ahora vamos a ir directamente al *Tercer Paso*, el último *paso* en este libro (un paso corto en longitud de páginas del libro, pero un paso que puede extenderse años en tu futuro) y vamos a ver cómo podés armar tu propia rutina de aprendizaje para convertirte en una persona experta en IA y definitivamente ¡transformar tu futuro!

"Este es un pequeño paso en el libro, pero un gran salto para tu futuro en la IA."



Tercer Paso: Formarte en base a la RUTINA



Formándote en Inteligencia Artificial: Una Guía Práctica

Ahora que ya tenés una base sólida sobre qué es la IA es momento de dar el siguiente paso. Este capítulo te servirá como una guía práctica y concreta para iniciar tu viaje en el fascinante campo de la IA.

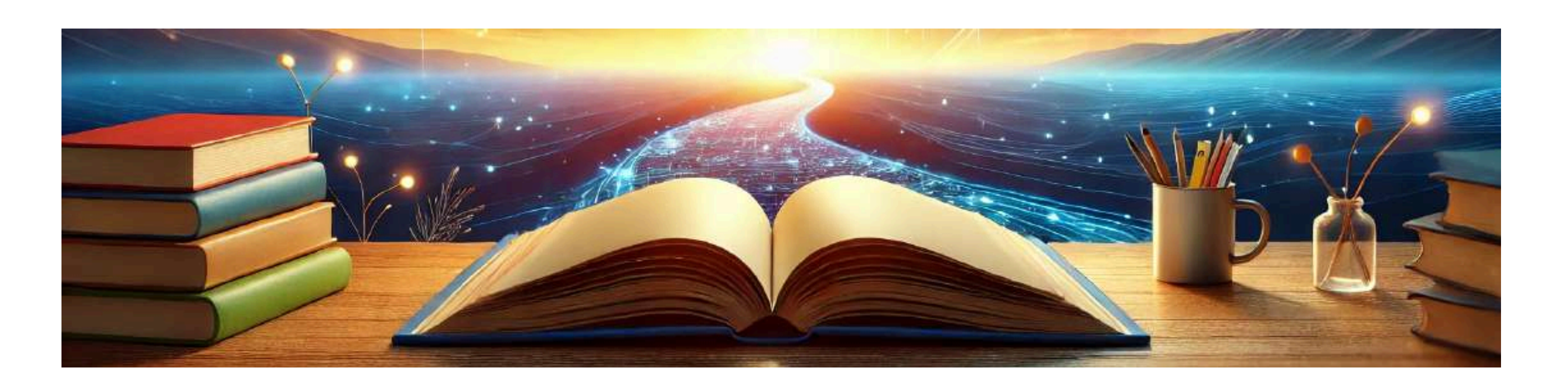
Ya sea que tu objetivo sea integrar la IA en tu trabajo, en tus proyectos personales o simplemente ampliar tus conocimientos, lo importante es comenzar con el pie derecho. Preparé una selección de recursos tanto en inglés como en español para que puedas aprovechar al máximo esta experiencia de aprendizaje.

Recordá, el camino del aprendizaje es personal y único. Vos decidís qué material aprovechar y cómo adaptarlo a tus necesidades. Te aseguro que, incluso con solo un par de los recursos que te voy a recomendar, vas a poder sumar una gran cantidad de herramientas y conocimientos de IA a tu vida.

Llegados a este punto, quiero decirte que tu aprendizaje no se detiene con este libro, sino que esto es solo un punto de partida. Prometí darte nociones básicas para que tengas un entendimiento de toda la disciplina de la inteligencia artificial y ganar contexto de todo lo que abarca. ¡Espero haberlo cumplido!

Lógicamente que para especializarse y aprender cómo aplicar todo esto en tu vida y fundamentalmente en tu ambiente laboral será cuestión de estudiar, practicar y perseverar, como casi todas las cosas difíciles de la vida. Pero la buena noticia es que hoy en día existen tantos recursos disponibles que se pueden aprovechar (muchos de ellos gratis) que no hay excusas para la persona que quiera realmente transformarse en especialista.

Mi recomendación principal será que establezcas una rutina de aprendizaje consistente para terminar de dominar la inteligencia artificial. La clave siempre está en la práctica diaria, aunque sea tan solo por momentos cortos. Dedicá tiempo cada día para estudiar conceptos leyendo artículos o viendo videos mientras tomás un café o hacés una pausa. Esta constancia no solo va a reforzar tu entendimiento de los temas, sino que también te va a ayudar a mantenerte al corriente en un campo que evoluciona rápidamente. Establecé metas realistas, celebrá tus pequeños logros y no te desanimes ante los desafíos.



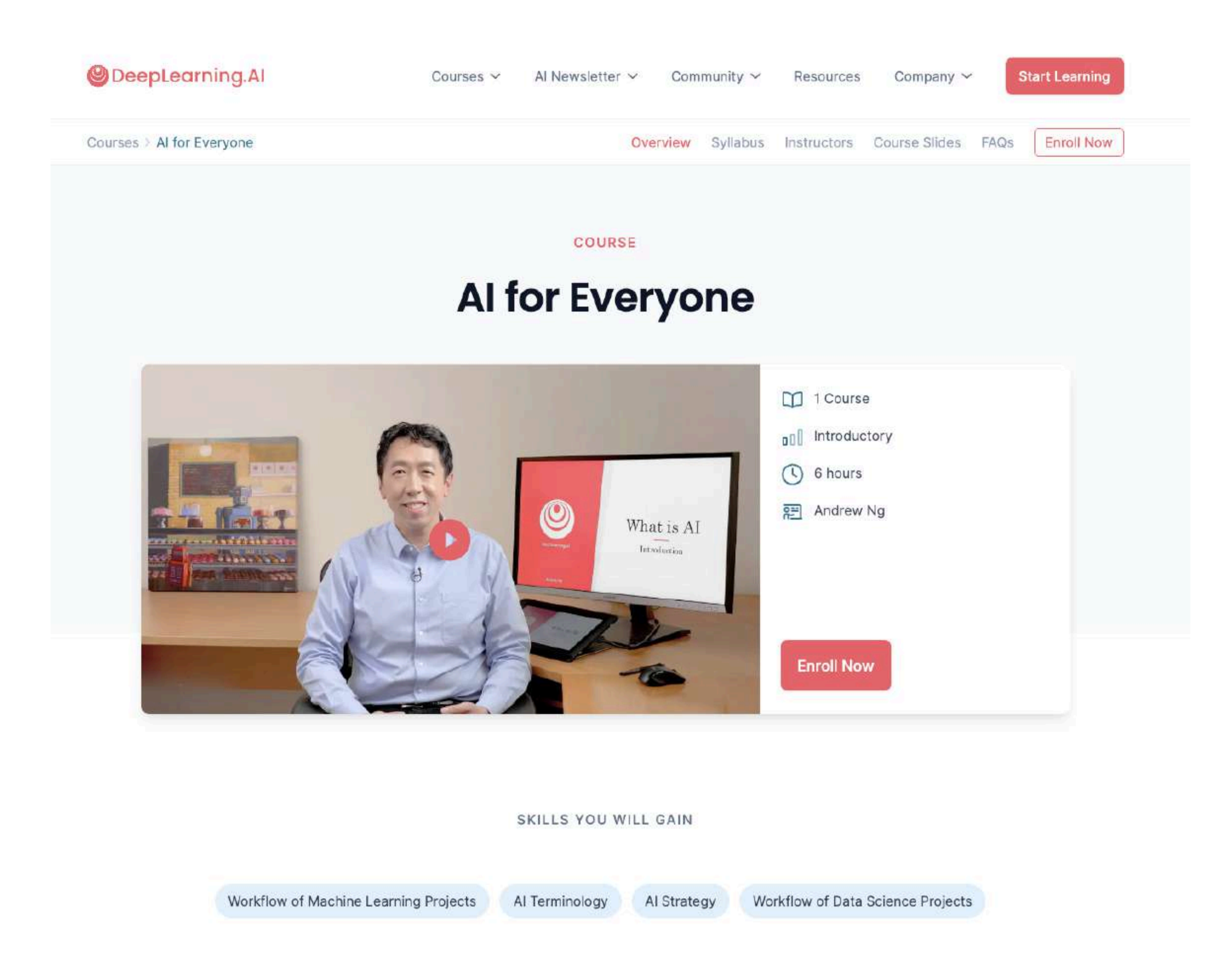
Antes de continuar quiero ser muy claro en algo. La calidad y variedad de material que se encuentra en idioma inglés, excede por mucho a lo que se encuentra en español, aunque si no te manejás con el inglés no te preocupes: por un lado, la mayoría de los cursos que te voy a recomendar en inglés tienen la opción de activar subtítulos en español. Por otro lado, estuve buscando directamente cursos en español en YouTube y encontré varios que me gustaron mucho y van a poder servir para nuestro propósito. Así que a partir de este punto, vos decidís qué parte de todo este material aprovechar y te aseguro que con solo recurrir a un par de ellos, vas a agregar un montón de herramientas y conocimientos de IA a tu vida y a tus habilidades.

Empecemos por uno de los mejores!

IA para tod@s

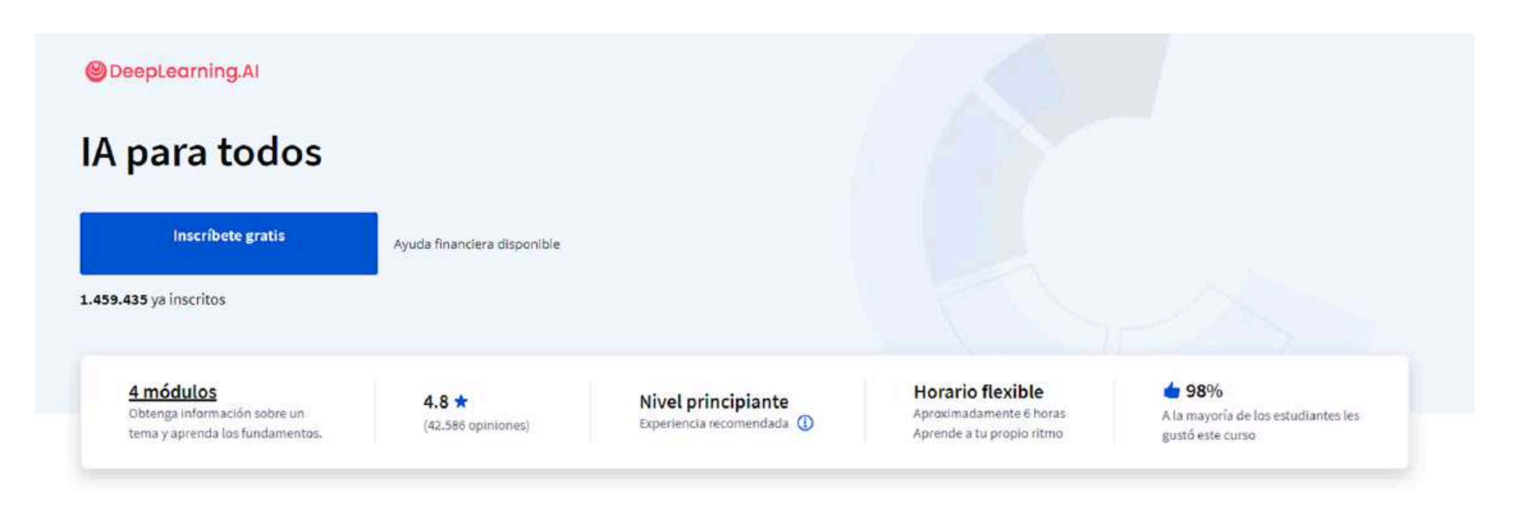
"AI for Everyone" de DeepLearning.AI
Precio: Gratuito
https://www.deeplearning.ai/courses/ai-for-everyone/.

Este curso es introductorio, está diseñado para personas no técnicas y proporciona una introducción accesible a la IA, su terminología, y cómo se puede aplicar en diferentes contextos. Es el curso perfecto si tu primer contacto con temas de IA fue este libro y querés terminar de entender perfectamente todo lo **básico.**

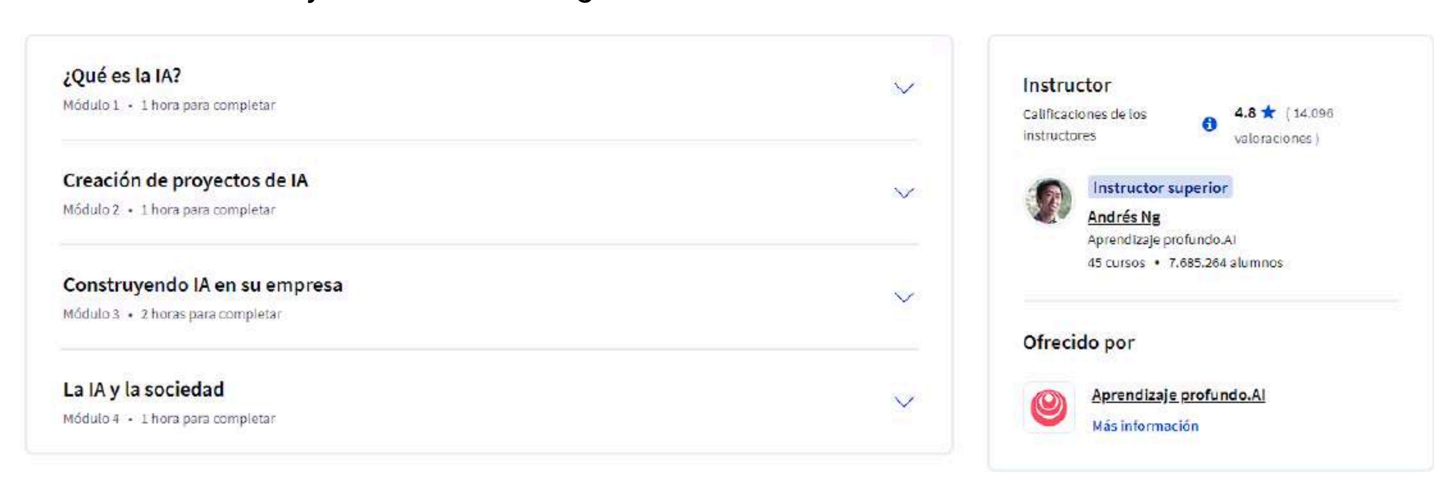


DeepLearning.AI es una empresa de tecnología educativa que busca empoderar a profesionales para construir un futuro impulsado por la inteligencia artificial. Fue fundada en

2017 por Andrew Ng, pionero en machine learning y educación, con la misión de ofrecer una educación de clase mundial y entrenamiento práctico.



El curso "AI for Everyone" cubre los siguientes temas:



🚀 **Módulo 1**: Introduce conceptos clave de la IA, qué es el *machine learning* y qué tipo de problemas puede (y no puede) resolver la IA. Además, brinda una explicación intuitiva del *deep learning* y discute las características clave de una empresa de IA.

🔍 **Módulo 2**: Se centra en el *workflow* de proyectos de *machine learning* y de ciencia de datos. Resalta cómo cada función laboral puede aprender a utilizar datos y cómo seleccionar proyectos de IA para tu equipo.

🎯 **Módulo 3**: Ofrece casos de uso sobre productos de IA como los altavoces inteligentes y los automóviles autónomos. Presenta roles típicos dentro de un equipo de IA y proporciona un manual para la transformación en IA, incluyendo trampas comunes a evitar.

🌍 **Módulo 4**: Examina el impacto de la IA en la sociedad, abordando temas como el sesgo y la discriminación, los ataques adversarios, y el uso adverso de la IA. También analiza cómo la IA podría afectar los empleos y el papel que puede jugar en los países en desarrollo.

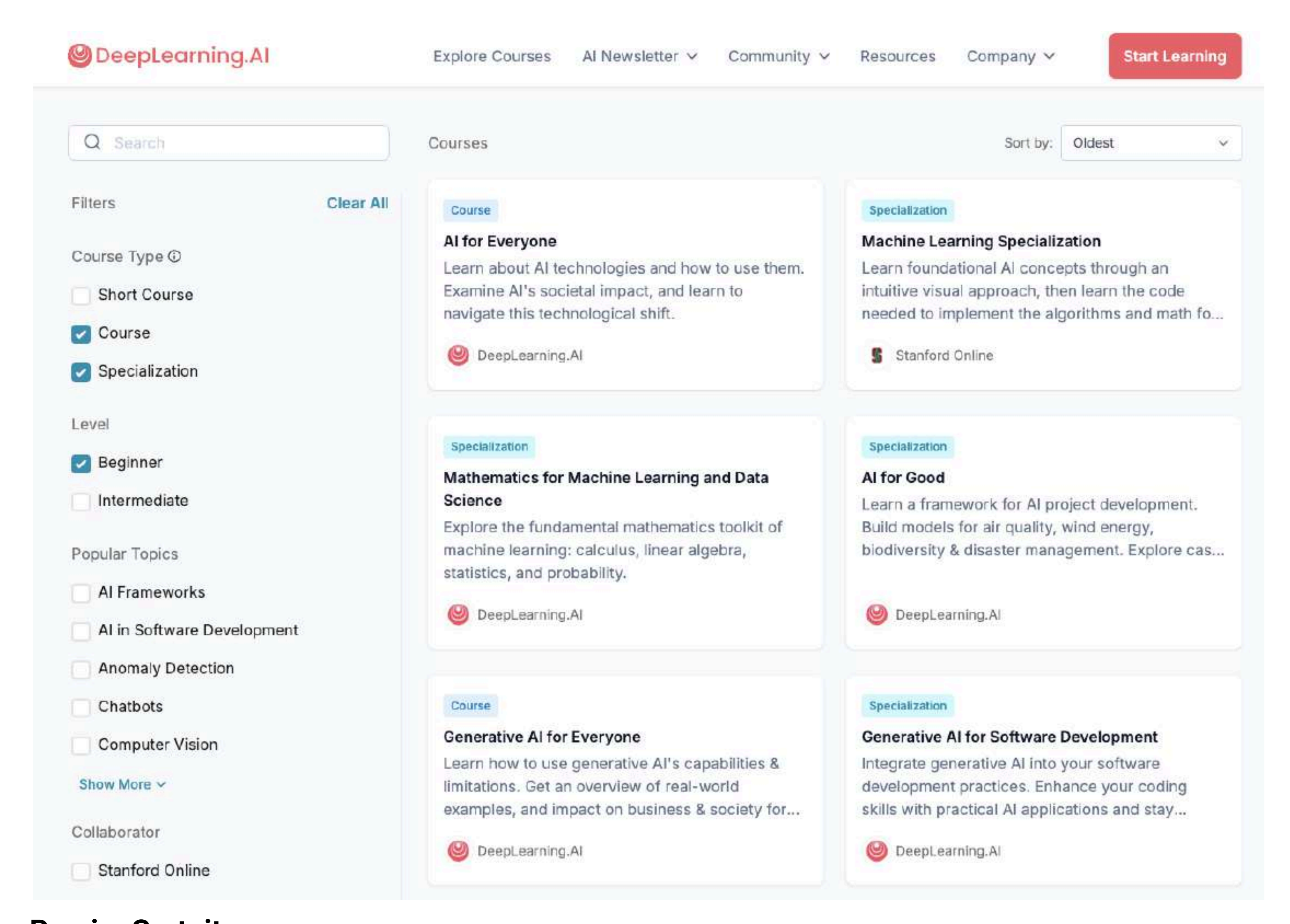
Tomando este curso como punto de partida, el siguiente paso es explorar cómo la IA puede encajar específicamente en tu ámbito de trabajo o interés personal. El enfoque del curso hacia una comprensión práctica y aplicada de la IA lo hace un excelente recurso para comenzar a planificar cómo podrías implementar o interactuar con la IA en tu entorno.

Entonces, con esta nueva comprensión y las herramientas proporcionadas por el curso, vas a tener todo para identificar oportunidades, planificar proyectos y participar en los debates sobre la IA de manera más informada.

Por dónde seguir

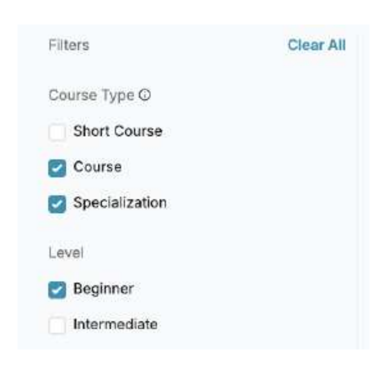
Una vez que hayas completado el curso "AI for Everyone" y tengas una base sólida, vas a poder explorar otros cursos introductorios que te van a permitir profundizar en diferentes aspectos de la inteligencia artificial y la ciencia de datos.

Podés encontrar estos cursos y muchos más en https://www.deeplearning.ai/courses/



Precio: Gratuitos

Para filtrar los mismos cursos que te muestro en la imágen anterior, seleccioná “Course”, “Specialization” y “Beginner” (o usa este link) en los filtros y elegí el curso que más se adecúa a tus objetivos.



Acá tenés una idea general de lo que podés esperar de algunos de los otros cursos básicos ofrecidos:

🤖 **Especialización en *machine learning***: Este curso está diseñado para proporcionar una comprensión más técnica del *machine learning*. Vas a aprender desde la teoría hasta la aplicación práctica de algoritmos, lo que te va a permitir construir y aplicar modelos predictivos. El contenido abarca tres partes fundamentales: primero enseña los conceptos básicos de aprendizaje supervisado, centrándose en regresión lineal y clasificación con regresión logística usando Python[131] y sus librerías NumPy y scikit-learn; luego avanza hacia algoritmos más avanzados; y finalmente cubre el aprendizaje no supervisado y sistemas de recomendación, todo orientado a desarrollar habilidades prácticas en inteligencia artificial y deep learning.

📊 **Especialización en matemáticas para *machine learning* y *data science***: Si en este libro te convencí de la importancia de las matemáticas para la IA y por lo tanto te interesa entender más sobre las bases que sustentan los algoritmos de IA, este curso te va a dar una sólida comprensión de conceptos clave como álgebra lineal, el cálculo y las estadísticas, esenciales para cualquier científico de datos o ingeniero de *machine learning*.

🌱 **IA para el bien (AI for Good)**: Estos cursos están enfocados en cómo aplicar la IA para resolver problemas sociales, ambientales y humanitarios. Ideal si estás interesado en cómo la tecnología puede hacer una diferencia positiva en el mundo.

[131] Si querés aprender Python desde cero, este curso de 4 horas dictado por Andrew Ng te enseña programación práctica con IA desde el primer día, usando un chatbot como asistente que te ayuda mientras construís aplicaciones reales como generadores de recetas y planificadores de viajes. El programa cubre los fundamentos de Python (variables, funciones, estructuras de datos) junto con el uso de librerías para análisis de datos y APIs, siendo ideal tanto si nunca programaste como si sos un profesional buscando integrar IA en tu trabajo. (https://www.deeplearning.ai/short-courses/ai-python-for-beginners/)

🎨 **IA generativa para tod@s** (**Generative AI for everyone)**: Este curso puede ser tu próximo paso si estás fascinado/a por la capacidad de la IA para generar nuevo contenido, ya sea texto, imágenes o música. Acá vas a explorar cómo funcionan los modelos generativos y cuáles son sus aplicaciones prácticas.

Cada uno de estos cursos introductorios está pensado para brindarte las herramientas y el conocimiento necesario para entender y aplicar la IA en una variedad de contextos, sin la necesidad de una experiencia técnica previa. Te van a equipar no solo con el conocimiento técnico, sino también con una perspectiva sobre cómo la IA se aplica en el mundo real y los principios éticos que deberían guiar su uso.

Al completar estos cursos, vas a estar en una posición mucho más fuerte para decidir cómo querés que la IA forme parte de tu carrera o vida personal, y finalmente estos cursos también establecen las bases que te van a permitir avanzar hacia contenidos más avanzados o especializados, dependiendo de tus intereses y objetivos profesionales.

Más recursos para continuar tu aprendizaje en IA

DeepLearning.AI ofrece un extenso catálogo de recursos, pero existen otras joyas en internet que considero importante compartir. A continuación, te presento una selección de cursos y videos que recopilé para complementar tu formación en inteligencia artificial y *machine learning*, abarcando desde conceptos básicos hasta técnicas avanzadas. Para quienes manejan el inglés, recomiendo especialmente los cursos gratuitos de instituciones prestigiosas como Stanford, IBM y Harvard, que incluyen subtítulos en español para facilitar la comprensión. Si el inglés no es lo tuyo, encontré excelentes playlists en YouTube completamente en español que cubren desde fundamentos hasta temas avanzados.

Es importante mencionar que no hace falta que uses todos estos materiales. La diversidad de recursos te permite elegir los que mejor se ajusten a tus intereses y nivel de experiencia. Algunos cursos incluyen programación en Python, ideales para quienes quieran aprender este lenguaje de programación. Todos estos recursos están pensados con un enfoque didáctico que combina teoría y práctica, permitiéndote aplicar lo que aprendiste en proyectos reales. Con esta variedad de opciones, vas a poder encontrar el contenido que mejor se adapte a tus necesidades y avanzar en tu aprendizaje de inteligencia artificial y *machine learning*.

Si no podés copiar los links de forma sencilla de la página siguiente, te sugiero que hagas una búsqueda en Google utilizando el "Nombre del curso/video" más el nombre de la "Plataforma" donde se publica.



Nombre del curso/video	Plataforma	Resumen del contenido del curso/video	Idioma audio	Subtit. Español?	Link
Machine Learning Specialization	Stanford University	La Especialización en machine learning de Stanford y DeepLearning.AI enseña fundamentos y aplicaciones prácticas de IA en tres cursos actualizados y ampliados.		Si	https://online.stanford.edu/courses/soe-ymls-machine-learning-specialization
Introduction to Artificial Intelligence (AI)	IBM	El curso "Introducción a la IA" de IBM en Coursera enseña fundamentos, aplicaciones, ética e impacto de la IA con laboratorios prácticos, cuestionarios y un certificado.		Si	https://www.coursera.org/learn/introduction-to-ai
HarvardX: CS50's Introduction to Artificial Intelligence with Python	HARVARD UNIVERSITY	Curso sobre conceptos y algoritmos de IA, con proyectos prácticos en Python, cubriendo machine learning, modelos de lenguaje y diseño de sistemas inteligentes.		Si	https://www.edx.org/learn/artificial-intelligence/harvard-university-cs50-s-introduction-to-artificial-intelligence-with-python
Descubriendo la IA	YouTube	Curso introductorio de inteligencia artificial que abarca fundamentos, aprendizaje supervisado y no supervisado, redes neuronales, NLP, visión por computadora y ética.		Si	https://youtube.com/playlist?list=PL_v-GoZ43zvZvpk3atOcd-nJ2F8qqxuWX&si=zt9YMaMZtCsRiR_i
¿Qué es machine learning?	YouTube	Este video explica cómo el machine learning permite a las máquinas aprender y mejorar su comportamiento, abordando aprendizaje supervisado, no supervisado y semi-supervisado.		Si	https://youtu.be/xrQ1YH0PnrM?si=kAYBJqhR2H6EV-rc
CURSO: MACHINE LEARNING CON PYTHON	YouTube	Curso de machine learning con Python que cubre desde fundamentos hasta técnicas avanzadas como clasificación, regresión, clustering y redes neuronales.		Si	https://www.youtube.com/playlist?list=PLJjOveEiVE4Dk48EI7I-67PEleEC5nxc3
Inteligencia Artificial y Machine Learning | Python	YouTube	Curso de Inteligencia Artificial y Machine Learning con Python, cubriendo conceptos fundamentales, algoritmos, aplicaciones prácticas y herramientas clave para desarrollar modelos.		Si	https://youtube.com/playlist?list=PLat2DtY8K7YWG4QxUruT1IB_sHumOr1Si&si=CdN6gOMU0W_ZLBvi
Curso Machine Learning con Python	YouTube	Curso de machine learning con Python que incluye fundamentos, técnicas avanzadas, implementación práctica y aplicaciones reales para principiantes y expertos.		Si	https://youtube.com/playlist?list=PL5C9QKu8AsmUDmAZNFiPEHxYTbZtliZZl&si=Kxl8cT0gDm7mo9No
Machine Learning español	YouTube	Curso de aprendizaje automático en español cubriendo desde fundamentos hasta técnicas avanzadas como regresión, clustering, redes neuronales y más.		Si	https://youtube.com/playlist?list=PLA050nq-BHwMr0uk7pPJUqRgKRRGhdvKb&si=OFOvhSKP_TsRWXa4

Esto concluye la sección de cursos y especializaciones, pero te quiero remarcar la importancia de hacer un aprendizaje continuo basado en la rutina. La clave del éxito en el campo de la inteligencia artificial no está solo en completar cursos, sino en mantener una práctica constante. Es preferible avanzar de a poco pero de manera sostenida, que intentar aprender todo de golpe y después abandonar por agotamiento. Además, el campo de la IA evoluciona tan rápido que mantenerse actualizado se volvió una parte fundamental del aprendizaje.

Aprendizaje continuo basado en la rutina



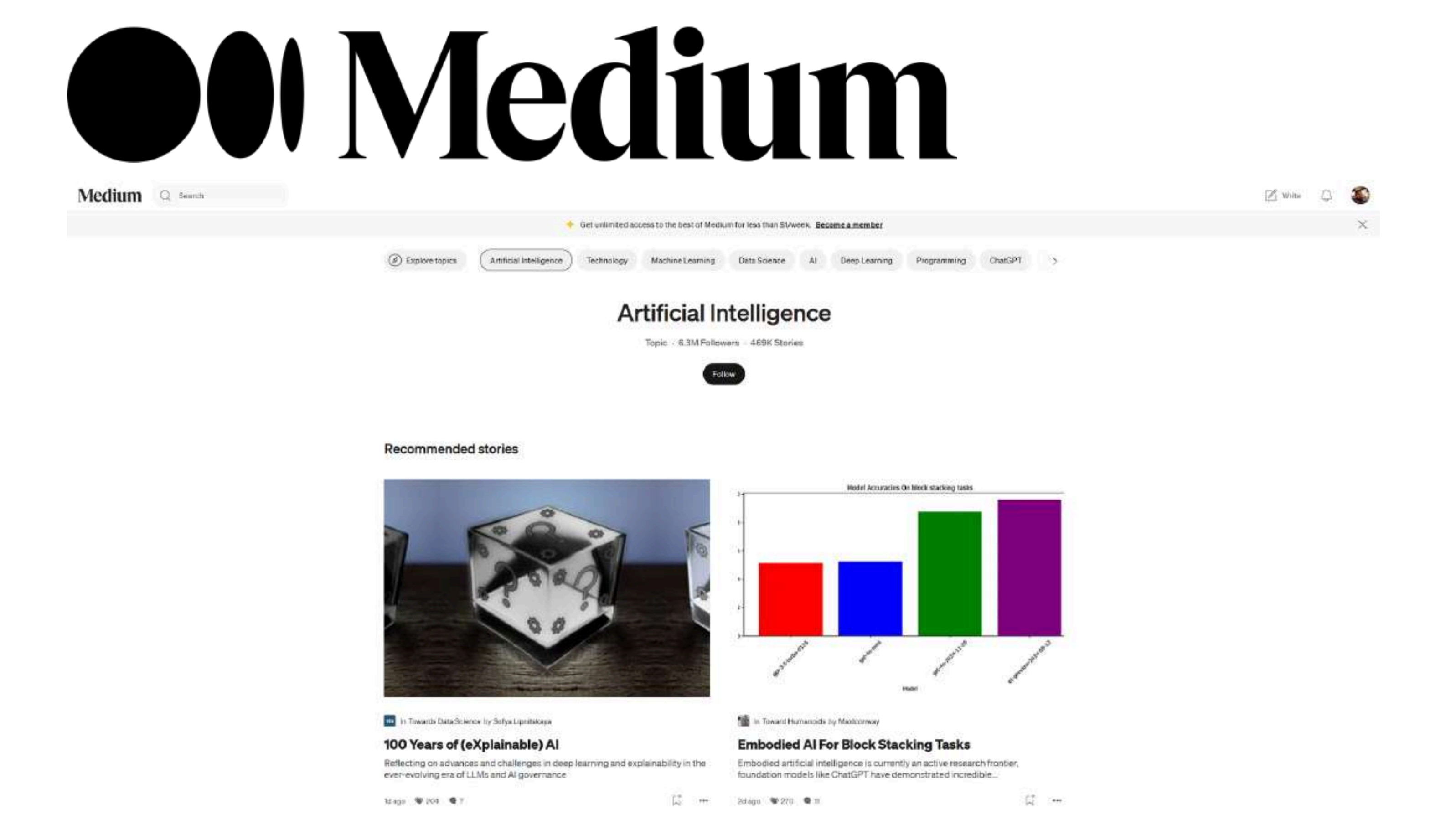
132

Tan importante como realizar cursos largos es que, día tras día, inviertas algunos minutos en leer algún artículo o mirar un video de algún canal de YouTube sobre inteligencia artificial. Si lográs adoptar este hábito creeme que vas a conseguir en muy poco tiempo ese *contexto* del que te hablé al comenzar este libro. Estoy seguro que sabés lo fácil que es olvidar un tema si no se lo practica suficiente y sería una pena que después de haber invertido este tiempo que dedicaste a leer este libro pasen semanas o meses y olvides los conceptos básicos. Por esto mismo, te sugiero añadir a la lista de tus páginas web favoritas los siguientes portales:

[132] Aprovecho para compartirte la playlist que usé todo el año mientras escribía este libro. Es ideal para concentrarse: https://www.youtube.com/watch?v=4xDzrJKXOOY&ab_channel=LofiGirl

Medium: Un tesoro de conocimiento en IA y ciencia de datos

https://medium.com/tag/artificial-intelligence



Medium es una plataforma de publicación digital que revolucionó la forma en que compartimos y consumimos contenido en línea. Fundada en 2012 por Evan Williams, cofundador de Twitter, Medium se convirtió en un espacio donde escritores, periodistas, expertos y aficionados pueden publicar artículos sobre una amplia gama de temas. La plataforma se destaca por su diseño limpio y enfocado en el contenido, permitiendo a los lectores sumergirse en historias, opiniones y análisis sin distracciones. Con un modelo que combina contenido gratuito y de pago, Medium ofrece a los creadores la oportunidad de monetizar su trabajo mientras brinda a los lectores acceso a contenido de calidad. Su sistema de recomendación y selección ayuda a los usuarios a descubrir nuevos artículos y autores, fomentando una comunidad diversa y dinámica de escritores y lectores. En este caso te sugiero su portal de noticias sobre *inteligencia artificial* y recordá que si no te manejás con inglés podés usar tu navegador para traducir todo el contenido al español.

Medium, por otro lado, alberga "Towards Data Science", un portal dedicado exclusivamente a explorar las últimas tendencias, técnicas y avances en ciencia de datos.

Towards Data Science: explorando las fronteras del conocimiento

https://towardsdatascience.com/

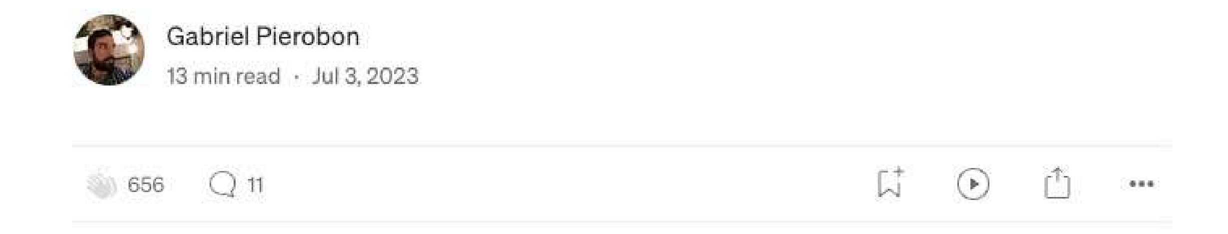
towards
data science

"Towards Data Science" se ha establecido como una comunidad central para expertos y aficionados en los campos de la IA y la ciencia de datos. Este portal ofrece una variedad de artículos escritos por profesionales y académicos que abordan desde conceptos básicos hasta temas avanzados.

Member-only story



Understanding Entropy made me a better data scientist



Gabriel Pierobon
13 min read · Jul 3, 2023

656 11

I remember several years ago when I was reshaping my career from finance into data science and being fascinated about how the book *Data Science for Business (Provost & Fawcett)* introduced the concept of ***Entropy*** in their classification examples, so elegantly, so powerful yet so simple. What they were explaining was nothing new to me, I had learned about machine learning and data science way before reading that book, yet that specific approach changed my whole interpretation of the subject. I always thought it was something truly beautiful to write about, hence this small article! Let's do it!

Incorporar a "Towards Data Science" en tu rutina de aprendizaje no solo te va a proporcionar acceso a información de vanguardia, sino que también te va a permitir explorar cómo se aplican estas tecnologías en diferentes industrias y contextos profesionales.

Para aprovechar al máximo los recursos disponibles en Medium y Towards Data Science, considerá los siguientes pasos:

1. **Lectura diaria**: Dedicá tiempo cada día para leer al menos un artículo del portal. Esto te va a ayudar a construir y mantener una base sólida de conocimientos en IA y ciencia de datos.
2. **Participá en la comunidad**: Enganchate con la comunidad. Comentá en los artículos, preguntá dudas y compartí tus propias experiencias. La interacción con otros lectores y autores puede proporcionar nuevas perspectivas y aclarar conceptos complejos.
3. **Aplicá lo aprendido**: Intentá implementar algunos de los métodos y técnicas que descubras en tus propios proyectos. La aplicación práctica es una de las formas más efectivas de consolidar el conocimiento y fomentar la innovación personal.

Estamos llegando al final del libro y no voy a omitir tal vez el mayor portal de aprendizaje disponible. Nunca subestimes la capacidad de YouTube de brindarte un contenido de primerísimo nivel.

YouTube: Una plataforma dinámica para el aprendizaje en IA



YouTube no es solo un lugar para entretenimiento. Es también una fuente invaluable de educación, especialmente en áreas técnicas como la inteligencia artificial (IA) y la ciencia de datos. Con acceso a expertos de todo el mundo, YouTube ofrece una variedad de canales educativos que proporcionan lecciones detalladas y tutoriales en un formato accesible y visual.

No voy a abusar de recomendarte canales porque es un proceso de descubrimiento de carácter más personal y ajustado a las preferencias de cada uno, pero voy a darte los que creo que si o si tenés que considerar.

Mis canales recomendados

1. StatQuest con Josh Starmer:
https://www.youtube.com/@statquest



StatQuest es un canal esencial para aquellos que desean desmitificar las estadísticas y el *machine learning*. Josh Starmer, el creador del canal, explica conceptos complejos de una manera simple y clara, utilizando ejemplos visuales que facilitan el aprendizaje de temas que pueden ser intimidantes al principio. Es ideal para principiantes y para aquellos que buscan solidificar su entendimiento de las estadísticas aplicadas a la ciencia de datos.

Para empezar te sugiero esta playlist de *machine learning:*

https://www.youtube.com/playlist?list=PLblh5JKOoLUICTaGLRoHQDuF_7q2GfuJF

2. CrashCourse:
https://www.youtube.com/@crashcourse



Conocido por sus cursos compactos y bien estructurados, **CrashCourse** ofrece series sobre una amplia gama de temas, incluyendo ciencia de la computación e inteligencia artificial. Estos cursos son perfectos para estudiantes y profesionales que buscan obtener una visión general rápida pero comprensiva de un tema.

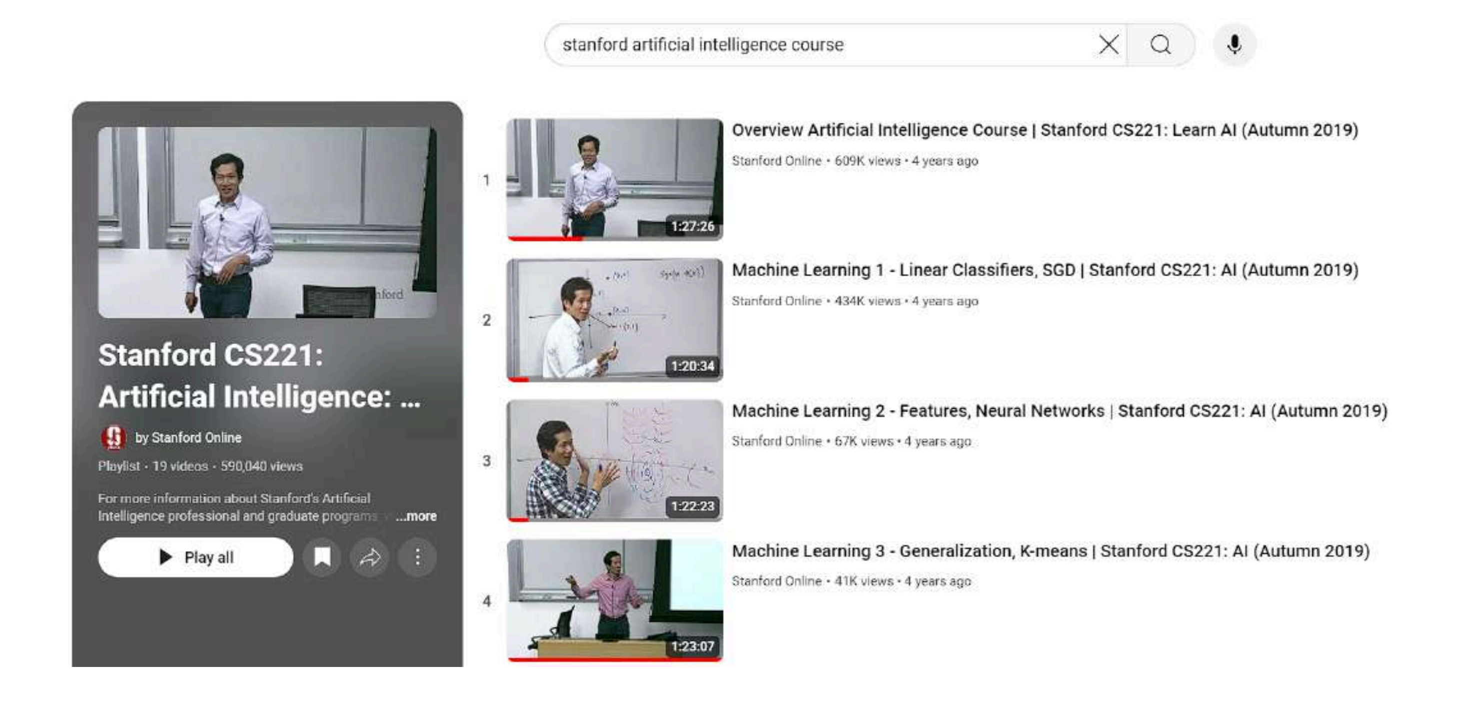
Mi favorito para que empieces, un curso completo sobre Inteligencia Artificial:
https://youtube.com/playlist?list=PLH2l6uzC4UEVGUu2--3xBjTMFily1IwP9&si=injs-fOnCxCh1qE9

3. Stanford Online:
https://www.youtube.com/@stanfordonline



El canal de **Stanford Online** es un tesoro de conocimientos académicos, ofreciendo acceso gratuito a cursos completos sobre temas avanzados. Ahí podés encontrar series de conferencias sobre IA, robótica, y más, dictadas por líderes en el campo. Es una opción excelente para aquellos que buscan una educación de nivel universitario sin costo alguno. Ciertamente son videos y cursos que van desde primeros años de universidad hasta maestrías y doctorados. Lamentablemente son 100% en inglés.

Mi curso favorito de Stanford Online:
https://www.youtube.com/playlist?list=PLoROMvodv4rO1NB9TD4iUZ3qghGEGtqNX



En resumen: tus próximos pasos

Para continuar tu camino en el aprendizaje de la IA, te dejo las recomendaciones clave:

Cursos Fundamentales

- Empezá con "AI for Everyone" de DeepLearning.AI (gratuito)
- Explorá las especializaciones básicas según tu interés:
  - Machine Learning
  - Matemáticas para IA
  - IA Generativa
  - IA para el Bien

Recursos de Aprendizaje Continuo

- Medium y Towards Data Science para artículos actualizados
- Canales de YouTube recomendados:
  - StatQuest con Josh Starmer
  - CrashCourse
  - Stanford Online

Esto nos lleva a concluir el *Tercer Paso* y dejame compartir una reflexión importante: aprender sobre inteligencia artificial no es una carrera corta, es más bien como prepararse para una maratón. Como pudiste ver a lo largo de este último paso, hay muchísimos recursos valiosos disponibles, tanto en inglés como en español, desde niveles básicos hasta más avanzados. La clave está en que armes tu propia rutina de aprendizaje, algo que puedas mantener en el tiempo de manera consistente. No es necesario consumir todo el material de una vez, al contrario, lo más efectivo es avanzar de manera constante y sostenida. Pueden ser quince minutos leyendo un artículo en Medium mientras tomás el café de la mañana, un video educativo de YouTube durante el almuerzo, o dedicar un momento los fines de semana a completar un módulo de algún curso. Lo importante es encontrar un ritmo que te resulte cómodo y puedas mantener.

Todos los recursos que te compartí son diferentes puertas de entrada al fascinante mundo de la IA. No es necesario explorarlos todos. Te sugiero elegir los que mejor se adapten a tus objetivos y tu situación actual.

Se sabe que la IA avanza a pasos agigantados, y mantenerse actualizado puede parecer un desafío importante. Sin embargo, con las bases que construiste a lo largo de este libro y los recursos que ahora tenés a tu disposición, contás con las herramientas necesarias para seguir creciendo en este campo.

Lo más valioso no va a ser la cantidad de cursos completados o artículos leídos, sino cómo aplicás estos conocimientos para resolver problemas reales y crecer profesionalmente. Cada paso que des en tu rutina de aprendizaje te acerca más a convertirte en un profesional del campo de la Inteligencia Artificial!

Nunca te olvides y lo voy a repetir todas las veces que sea necesario ;)

- 🎯 Establecé objetivos realistas y alcanzables. Es importante definir metas que te motiven pero que no te abrumen en el proceso.
- ⏰ Mantené una rutina consistente que puedas sostener. La regularidad es más valiosa que las sesiones intensivas esporádicas.
- 📚 Combiná diferentes recursos y formatos de aprendizaje. La variedad te ayudará a mantener el interés y abordar el conocimiento desde distintos ángulos.
- 💻 Practicá lo aprendido en proyectos concretos. La aplicación práctica es fundamental para consolidar tus conocimientos.
- 🌟 Celebrá tus logros, por pequeños que parezcan. Reconocer tu progreso te mantendrá motivado para seguir avanzando.
- 🚀 Y recordá que cada obstáculo es una oportunidad de aprendizaje. Los desafíos son parte natural del proceso de crecimiento.

Y recordá que para progresar hace falta:

- Voluntad para dar el primer paso
- Labor constante día a día
- Logro de pequeñas metas
- Constancia hasta el final



¡Y es así como llegamos también al final del libro!

❤️ Muchas gracias de corazón por haberlo leído parcial o completamente.

🥺 Me emociona escribir esta página pero tampoco quiero extenderme mucho más, por que esto se trata de **tu** aprendizaje. Y no tiene que detenerse acá. 🚀

Espero haber cumplido con mi promesa de aportarte *contexto* y tus primeras herramientas para abordar el desafiante mundo de la inteligencia artificial. Ojalá que puedas considerar este libro como un material de consulta por muchos años y si algún día considerás que te sirvió para complementar tu carrera, para conseguir un trabajo que deseabas o simplemente para tener más conocimiento, me encantaría que me lo cuentes. Te invito a que sigamos en contacto en linkedin https://www.linkedin.com/in/gabrielpierobon/ y que me hagas llegar tus dudas, comentarios, sugerencias o consultas por esa plataforma si así lo necesitás.

¡Te deseo muchos éxitos personales y profesionales!

Gabriel Pierobon

El contenido de este libro, y fundamentalmente lo que se encuentra dentro de recuadros fue desarrollado con asistencia de modelos de inteligencia artificial, específicamente Claude de Anthropic, y editado por el autor entre los meses de abril de 2024 y febrero de 2025. Las imágenes, salvo que se indique lo contrario, fueron creadas utilizando DALL-E de OpenAI. Todo el contenido fue supervisado, revisado y aprobado por el autor. Este libro ha sido creado con fines educativos y de divulgación, sin ningún tipo de ánimo de lucro. A través de estas páginas, el autor ha querido ofrecer su propia visión del presente y futuro de la Inteligencia Artificial, entretejiendo conceptos técnicos con las experiencias y aprendizajes de su trayectoria profesional y sus investigaciones en el campo. El objetivo del autor no es posicionarse como un experto técnico, sino servir de puente para que conceptos complejos sean más accesibles y comprensibles para todos aquellos interesados en adentrarse en el fascinante mundo de la IA. Habiendo identificado una escasez de material comprehensivo sobre este tema en español, el autor busca hacer este conocimiento accesible a aquellas personas que, no dominando el inglés, se han visto limitadas en su acceso a recursos de calidad sobre Inteligencia Artificial.

Título del Libro: "Inteligencia Artificial: Los 3 pasos para formarte desde cero y transformar tu futuro"

Autor: Gabriel Pierobon

Este libro no pretende ser un compendio exhaustivo de todos los conceptos y técnicas de la Inteligencia Artificial, sino una guía práctica y accesible que cubre los fundamentos esenciales. Está especialmente diseñado tanto para profesionales que buscan actualizarse en el campo de la IA como para personas que están dando sus primeros pasos en esta disciplina. El enfoque elegido prioriza la claridad y la aplicabilidad práctica por sobre la exhaustividad técnica, proporcionando una base sólida para que el lector pueda seguir profundizando en las áreas que más le interesen.

Primera edición: v1.0.0 02/2025

Publicado por Gabriel Pierobon

Madrid, España

